

LAPORAN TUGAS BESAR

SISTEM E-COMMERCE



PENYUSUN

- (11S19055) (Kartika Hutaeruk)
- (11S22011) (Christian Napitupulu)
- (11S22023) (Yessi Sitanggang)
- (11S22033) (Chalvin Sihombing)
- (11S22043) (Esra Silaen)
- (11S22054) (Carlos Pardomuan Purba)

PROGRAM STUDI SARJANA INFORMATIKA
FAKULTAS INFORMATIKA DAN TEKNIK ELEKTRO
INSTITUT TEKNOLOGI DEL
DESEMBER 2023

DAFTAR ISI

DAFTAR ISI.....	2
DAFTAR GAMBAR.....	5
DAFTAR TABEL.....	8
BAB 1 PROSES BISNIS	9
BAB 2 DIAGRAM BASIS DATA.....	10
2.1 ER Diagram.....	10
2.2 CDM & PDM.....	11
2.2.1 CDM	11
2.2.2 PDM.....	12
2.3 TABEL FISIK	13
BAB 3 NORMALISASI.....	19
3.1 Normalisasi Bentuk Pertama (1NF)	19
3.2 Normalisasi Bentuk Kedua (2NF).....	21
3.3 Normalisasi Bentuk Ketiga (3NF).....	24
BAB 4 IMPLEMENTASI.....	26
4.1 Implementasi Database ke SQL Server.....	26
4.2 Creating and Maintaining Database	26
4.2.1 Creating, Insert and View Maintaining Tabel.....	29
4.2.1.1 Creating Tabel Akun	29
4.2.1.2 Viewing Tabel Akun	30
4.2.1.3 Inserting Value Tabel Akun	31
4.2.1.4 Creating Tabel Pelanggan.....	33
4.2.1.5 Viewing Tabel Pelanggan.....	34
4.2.1.6 Inserting Tabel Pelanggan	35
4.2.1.7 Creating Tabel Order	36
4.2.1.8 Viewing Tabel Order	38
4.2.1.9 Inserting Tabel Order.....	38

4.2.1.10 Creating Tabel Pembayaran.....	40
4.2.1.11 Viewing Tabel Pembayaran.....	41
4.2.1.12 Inserting Tabel Pembayaran	41
4.2.1.13 Creating Tabel Detail Pesanan	43
4.2.1.14 Viewing Tabel Detail Pesanan	44
4.2.1.15 Inserting Tabel Detail Pesanan	44
4.2.1.16 Creating Tabel Penjual	46
4.2.1.17 Viewing Tabel Penjual	46
4.2.1.18 inserting Tabel Penjual	47
4.2.1.19 Creating Tabel Pengiriman.....	49
4.2.1.20 Viewing Tabel Pengiriman.....	49
4.2.1.21 Inserting Tabel Pengiriman	50
4.2.1.22 Creating Tabel Produk.....	52
4.2.1.23 Viewing Tabel Produk.....	52
4.2.1.24 Inserting Tabel Produk	53
4.2.1.25 Creating Tabel Keranjang.....	55
4.2.1.26 Viewing Tabel Keranjang.....	55
4.2.1.27 Inserting Tabel Keranjang	56
4.2.1.28 Creating Tabel Tinjauan	58
4.2.1.29 Viewing Tabel Tinjauan	59
4.2.1.30 Inserting Tabel Tinjauan.....	59
4.3 Creating Basic Query (Select, Update, Set Operators, Aggregate Function, Null Value)..	61
4.4 Quering Multiple Tabel.....	73
4.5 Nested Sub-Queries.....	77
4.6 Join Tabel	82
4.7 Creating Views.....	87
4.8 Function.....	93
4.9 Procedure.....	95
BAB 5 KESIMPULAN DAN SARAN	97
5.1 Kesimpulan.....	97
5.2 Saran.....	98

DAFTAR PUSTAKA	99
LAMPIRAN.....	100

DAFTAR GAMBAR

Gambar 1 ERD Sistem E-commerce	10
Gambar 2. CDM Sistem E-commerce	11
Gambar 3. PDM Sistem E-commerce.....	12
Gambar 4. Create Database Proyek_Basdat_Kelompok01	27
Gambar 5. Hasil Ouput Create Database Ecommerce	28
Gambar 6. Creating Tabel Akun	29
Gambar 7. Hasil Output Creating Tabel Akun	30
Gambar 8. Insert Value into Tabel Akun.....	31
Gambar 9. Hasil Output Insert Value into Tabel Akun	32
Gambar 10. Creating Tabel Pelanggan	33
Gambar 11. Hasil Output Creating Tabel Pelanggan.....	34
Gambar 12. Insert Value into Tabel Pelanggan	35
Gambar 13. Hasil Output Insert Value into Tabel Pelanggan	36
Gambar 14. Creating Tabel Orders	37
Gambar 15. Hasil Output Creating Tabel Orders	38
Gambar 16. Insert Value into Tabel Orders.....	39
Gambar 17. Hasil Output Insert Value into Tabel Orders	39
Gambar 18. Creating Tabel Pembayaran	40
Gambar 19. Hasil Output creating Tabel Pembayaran	41
Gambar 20. Insert Value into Tabel Pembayaran	42
Gambar 21. Hasil Output Insert Value into Tabel Pembayaran	42
Gambar 22. Creating Tabel Detail Pesanan	43
Gambar 23. Hasil Output Creating Tabel Detail Pesanan	44
Gambar 24. Insert Value into Tabel Detail Pesanan.....	45
Gambar 25. Hasil Output Insert Value into Tabel Detail Pesanan	45
Gambar 26. Creating Tabel Penjual.....	46
Gambar 27. Hasil Output Creating Tabel Penjual	47
Gambar 28. Insert Value into Tabel Penjual.....	48
Gambar 29. Hasil Ouput Insert Value into Tabel Penjual	48
Gambar 30. Creating Tabel Pengiriman	49
Gambar 31. Hasil Output Creating Tabel Pengiriman.....	50
Gambar 32. Insert Value into Tabel Pengiriman	51
Gambar 33. Hasil Output Insert Value into Tabel Pengiriman.....	51
Gambar 34. Creating Tabel Produk	52
Gambar 35. Hasil Output Creating Tabel Produk.....	53
Gambar 36. Insert Value into Tabel Produk	54
Gambar 37. Hasil Output Insert Value into Tabel Produk.....	54
Gambar 38. Creating Tabel Keranjang	55
Gambar 39. Hasil Output Creating Tabel Keranjang.....	56
Gambar 40. Insert Value into Tabel Keranjang	57
Gambar 41. Hasil Output Insert Value into Tabel Keranjang	57

Gambar 42. Creating Tabel Tinjauan.....	58
Gambar 43. Hasil Output Creating Tabel Tinjauan	59
Gambar 44. Insert Value intoTabel Tinjauan	60
Gambar 45. Hasil Output Insert Value intoTabel Tinjauan	60
Gambar 46. Melakukan Drop Tabel Akun	61
Gambar 47. Insert Value Data ke dalam Tabel Detail Pesanan	62
Gambar 48. Hasil Output Data yang Diinput ke dalam Tabel Detail Pesanan	63
Gambar 49. Hasil Execute Mengupdate Salah Satu Nama Pelanggan	64
Gambar 50. Displays Pesanan Item Data That Has an Nama Pesanan with the Final Letter 'G' and Harga Pesanan > 200,000	65
Gambar 51. Update Data From Table Detail Pesanan	66
Gambar 52. Add Aktivitas Column ke dalam Table Pelanggan	67
Gambar 53. Deleting Aktivitas Column Pada Table Pelanggan.....	68
Gambar 54. Penambahan Harga Sebesar Rp. 10.000 Pada Kolom Harga Satuan.....	69
Gambar 55. Perkalian antara Kolom kuantitas dengan Harga satuan untuk memperoleh Total Harga.....	70
Gambar 56. Menampilkan data Pesanan dari Detail Pesanan yang memiliki kuantitas di atas 3.	71
Gambar 57. Menampilkan Id_Produk dan Nama_Pesanan dari Tabel Produk	72
Gambar 58. Show Different Value dari Id_Produk dan Nama_Pesanan.....	73
Gambar 59. Show All No_Pesanan dan d_Produk dari Setiap Pemesanan yang dilakukan user.	74
Gambar 60. Show All Bagian yang berbeda dari semua Pesanan	75
Gambar 61. Pengelompokan Data Menjadi Satu Data yang unik dari Tabel Detail Pesanan	76
Gambar 62. Count Number Of Unique Rows dari Tabel Detail Pesanan	77
Gambar 63. Menampilkan Nama dan No_telepon dari Akun yang memiliki Email 'gmail.com'	78
Gambar 64. Menampilkan Nama dari Akun yang memiliki Id_Akun unik (tidak ada yang sama dengan akun lain).....	79
Gambar 65. Menampilkan Id_Produk dan Nama_Pesanan dari pesanan yang Status_Pesanan-nya sudah dibayar ('Sudah dibayar')	80
Gambar 66. Menampilkan No_Pesanan dan Tanggal_Pesanan dari pesanan dengan Kuantitas terbanyak.....	81
Gambar 67. Menampilkan No_Pesanan dan Id_Order dari pesanan yang menggunakan Jasa_Pengiriman 'JNT' dan belum dibayar	82
Gambar 68. INNER JOIN - Menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan.....	83
Gambar 69. LEFT JOIN - Menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan.....	84
Gambar 70. RIGHT JOIN - Menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan.....	85
Gambar 71. OUTER JOIN - Menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan.....	86
Gambar 72. CROSS JOIN - Menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan.....	87
Gambar 73. View 1 - Daftar Detail Pesanan.....	88
Gambar 74. View 2 - Daftar Pelanggan	89
Gambar 75. View 3 - Detail Order Pelanggan	90
Gambar 76. View 4 - Tinjauan Produk	91

Gambar 77. View 5 - Penjual dan Pengiriman	92
Gambar 78. Menghitung Total Harga Pesanan.....	93
Gambar 79. Menghitung Rata-rata Harga dari Produk.....	94
Gambar 80. Menampilkan Seluruh Data dari Tabel Order.....	95
Gambar 81. Menampilkan Seluruh Data dari Tabel Detail Pesanan	96

DAFTAR TABEL

Tabel 1. Akun.....	15
Tabel 2. Pelanggan.....	15
Tabel 3. Order	16
Tabel 4. Pembayaran.....	16
Tabel 5. Detail Pesanan.....	16
Tabel 6. Penjual	17
Tabel 7. Pengiriman.....	17
Tabel 8. Produk.....	17
Tabel 9. Keranjang.....	18
Tabel 10. Tinjauan	18
Tabel 11. Akun (1NF).....	19
Tabel 12. Pelanggan (1NF).....	19
Tabel 13. Order (1NF)	19
Tabel 14. Pelanggan (1NF).....	20
Tabel 15. Detail Pesanan (1NF).....	20
Tabel 16. Penjual (1NF).....	20
Tabel 17. Produk (1NF)	20
Tabel 18. Keranjang (1NF).....	20
Tabel 19. Tinjauan (1NF)	21
Tabel 20. Akun (2NF).....	21
Tabel 21. Penjual (2NF).....	22
Tabel 22. Order (2NF)	22
Tabel 23. Detail Pesanan (2NF).....	22
Tabel 24. Penjual (2NF).....	23
Tabel 25. Pengiriman (2NF)	23
Tabel 26. Produk (2NF)	23
Tabel 27. Keranjang (2NF)	23
Tabel 28. Tinjauan (2NF)	24
Tabel 29. Order (3NF)	24
Tabel 30. Pengiriman (3NF)	25
Tabel 31. Produk (3NF)	25

BAB 1

PROSES BISNIS

Proses bisnis ialah kegiatan terstruktur yang saling terkait untuk menyelesaikan suatu masalah tertentu atau yang menghasilkan sesuatu. Dalam laporan ini, proses bisnis dari sistem e-commerce seperti Shopee, Tokopedia, Lazada, dan sejenisnya melibatkan beberapa tahapan penting untuk memfasilitasi transaksi jual-beli antara penjual dan pembeli. Di mana sistem E-commerce ini juga merupakan platform yang memfasilitasi transaksi jual beli secara online antara penjual dan pembeli.

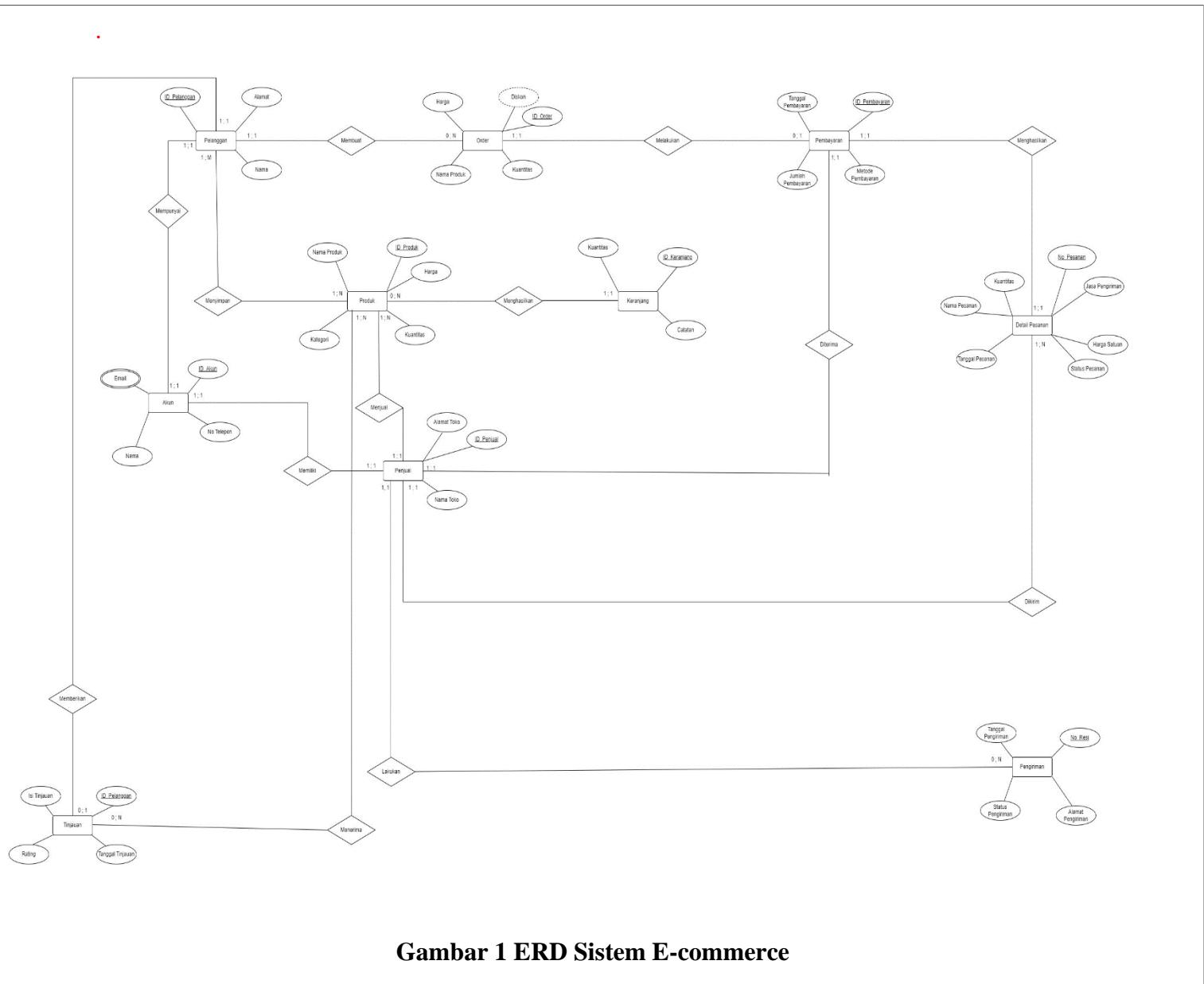
Proses bisnis dalam sebuah e-commerce dapat melibatkan langkah-langkah berikut:

1. Pendaftaran, di mana dalam hal ini pengguna dapat mendaftar sebagai anggota e-commerce dengan mengisi informasi pribadi dan membuat akun.
2. Pemilihan Produk. Dalam tahap ini pengguna dapat menjelajahi katalog produk yang tersedia dan memilih produk yang ingin dibeli.
3. Keranjang Belanja. Dalam model ini pengguna dapat menambahkan produk ke keranjang belanja mereka sebelum melanjutkan ke proses pembayaran.
4. Pembayaran. Dalam proses ini pengguna dapat memilih metode pembayaran yang diinginkan, seperti kartu kredit, transfer bank, atau metode pembayaran elektronik lainnya.
5. Setelah pembayaran berhasil, e-commerce akan mengatur pengiriman produk ke alamat yang telah ditentukan oleh pengguna.
6. Pelacakan Pesanan. Dalam proses ini pengguna dapat melacak status pengiriman pesanan mereka, biasanya melalui nomor pelacakan yang diberikan oleh e-commerce.
7. Pengembalian dan Penggantian. Jika ada masalah dengan produk yang diterima, pengguna dapat mengajukan permintaan pengembalian atau penggantian.
8. Setelah pengguna menerima pesanan mereka, mereka dapat memberikan ulasan dan penilaian terhadap produk dan pengalaman berbelanja mereka di e-commerce tersebut.

BAB 2

DIAGRAM BASIS DATA

2.1 ER Diagram

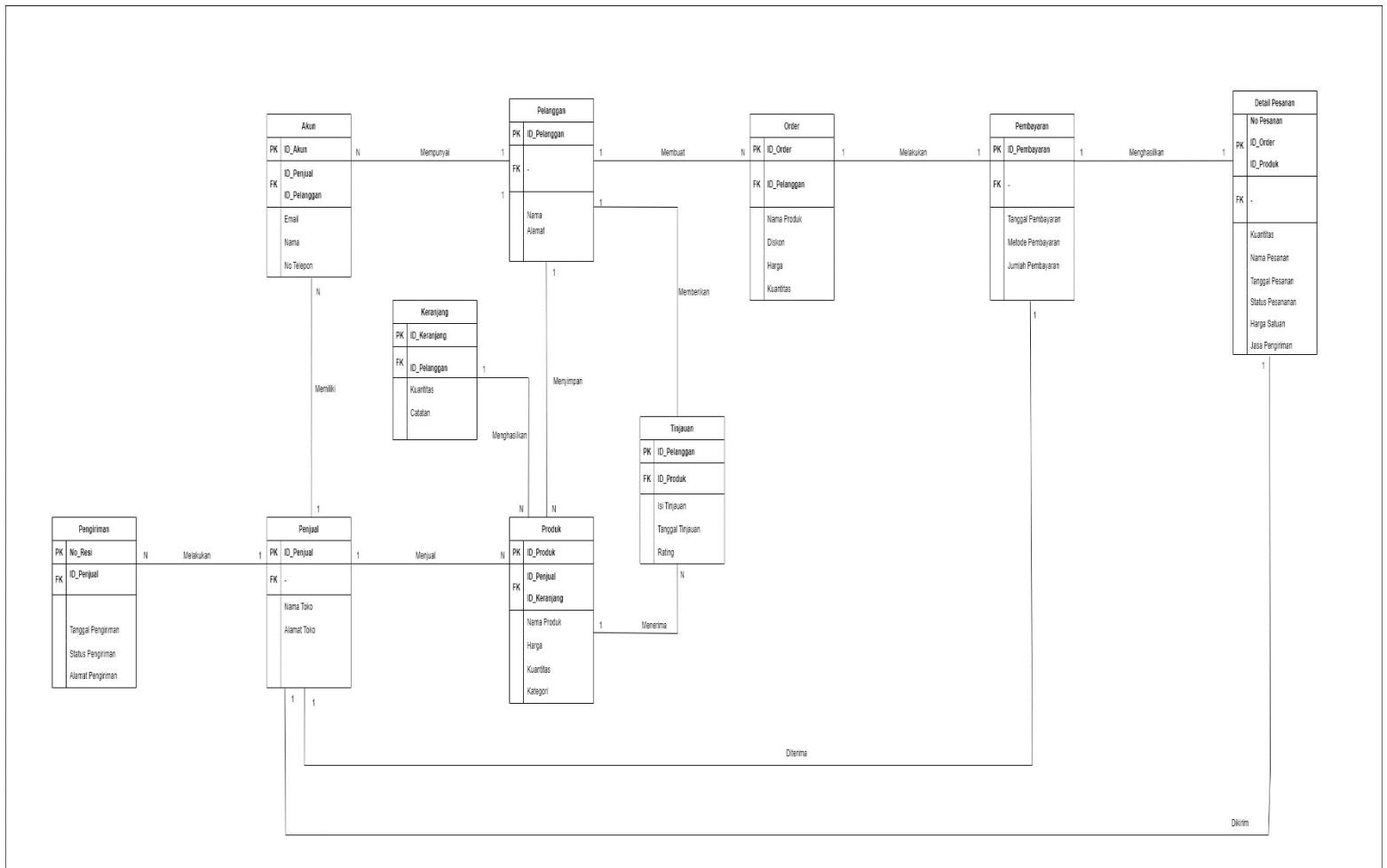


Gambar 1 ERD Sistem E-commerce

2.2 CDM & PDM

2.2.1 CDM

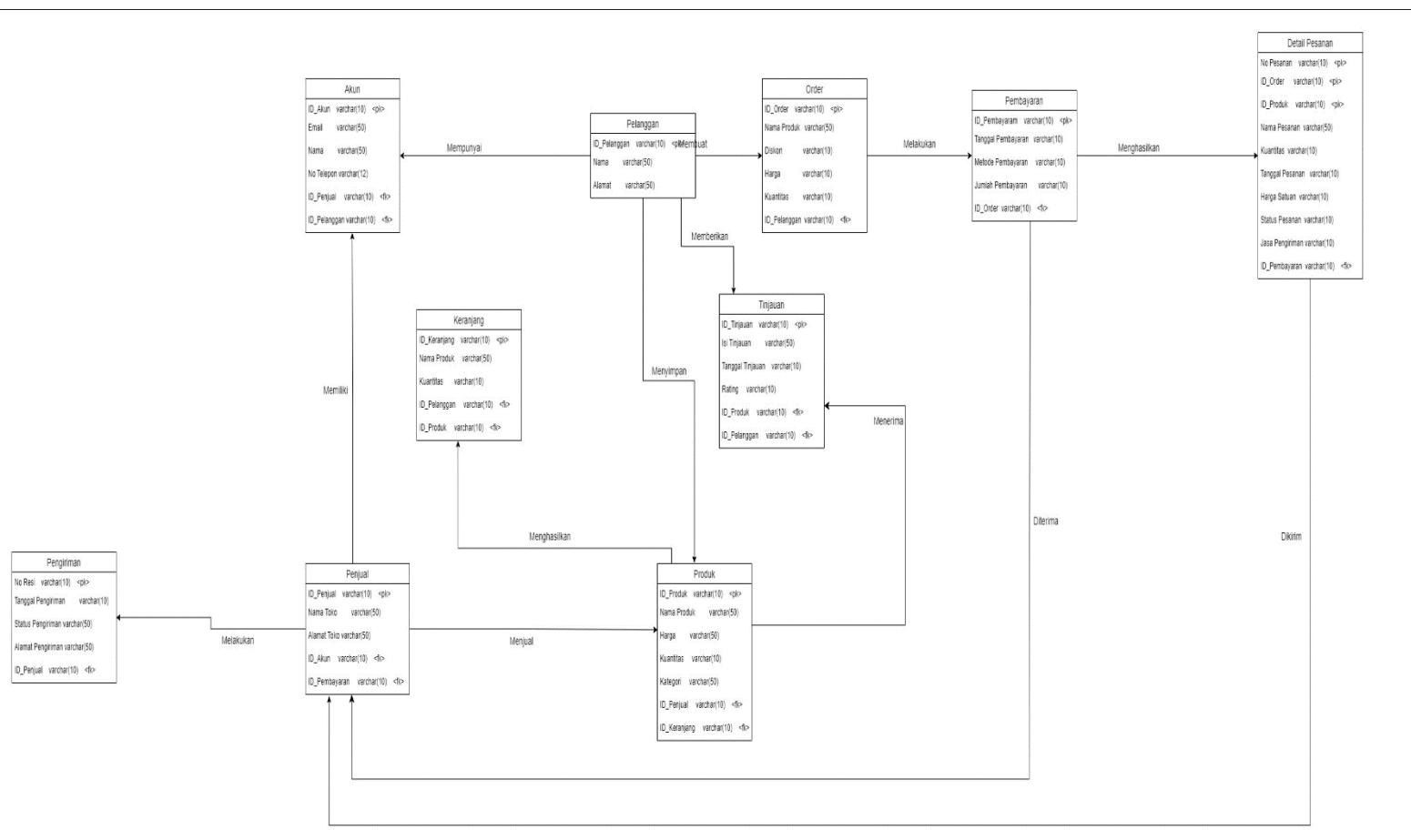
Conceptual Data Model (CDM) atau model konsep data merupakan konsep yang berkaitan dengan pandangan pemakai terhadap data yang disimpan dalam basis data. CDM dibuat sudah dalam bentuk tabel-tabel tanpa tipe data yang menggambarkan relasi antar tabel untuk keperluan implementasi ke basis data (Mariana, 2018). CDM merupakan hasil penjabaran lebih lanjut dari ERD. Berikut adalah CDM dari studi kasus Sistem E-Commerce:



Gambar 2. CDM Sistem E-commerce

2.2.2 PDM

Physical Data Model (PDM) atau Model relasional adalah model yang menggunakan sejumlah tabel untuk menggambarkan data serta hubungan antara data. Setiap tabel mempunyai sejumlah kolom dimana setiap kolom memiliki nama yang unik beserta tipe datanya. PDM merupakan konsep yang menerangkan detail dari bagaimana data disimpan di dalam basis data. PDM sudah merupakan bentuk fisik perancangan basis data yang sudah siap diimplementasikan ke dalam DBMS sehingga nama tabel juga sudah merupakan nama asli tabel yang akan diimplementasikan ke dalam DBMS (Mariana, 2018). Berikut adalah PDM dari studi kasus Sistem E-Commerce:



Gambar 3. PDM Sistem E-commerce

2.3 TABEL FISIK

Tabel fisik adalah representasi dari tabel di model data relasional yang disimpan di dalam basis data. Tabel fisik terdiri dari sejumlah baris dan kolom. Setiap baris mewakili satu entitas dalam basis data, dan setiap kolom mewakili satu atribut dari entitas tersebut.

A. Nama tabel

Nama tabel adalah unik dalam satu basis data. Nama tabel digunakan untuk mengidentifikasi tabel tersebut. Nama tabel biasanya berupa kata atau frasa yang menggambarkan data yang disimpan dalam tabel tersebut.

B. Nama kolom

Nama kolom juga harus unik dalam satu tabel. Nama kolom digunakan untuk mengidentifikasi kolom tersebut. Nama kolom biasanya berupa kata atau frasa yang menggambarkan atribut yang diwakili oleh kolom tersebut.

C. Tipe data kolom

Tipe data kolom menentukan jenis data yang dapat disimpan di dalam kolom tersebut. Tipe data kolom yang umum digunakan antara lain:

- Teks: Tipe data teks digunakan untuk menyimpan data berupa teks, seperti nama, alamat, dan keterangan.
- Numerik: Tipe data numerik digunakan untuk menyimpan data berupa angka, seperti angka umur, angka gaji, dan angka penjualan.
- Tanggal/Waktu: Tipe data tanggal/waktu digunakan untuk menyimpan data berupa tanggal dan waktu, seperti tanggal lahir, tanggal mulai kerja, dan tanggal transaksi.

D. Nilai kolom

Nilai kolom dapat berupa data teks, data numerik, atau data lain yang didukung oleh sistem basis data. Nilai kolom dapat dimasukkan ke dalam tabel melalui bahasa pemrograman atau aplikasi basis data.

E. Implementasi tabel fisik

Tabel fisik diimplementasikan sebagai sebuah file di penyimpanan permanen, seperti hard drive atau SSD. File ini berisi informasi tentang nama tabel, nama kolom, tipe data kolom, dan nilai-nilai yang disimpan di dalam tabel.

Tabel fisik digunakan untuk menyimpan data dalam basis data. Data yang disimpan dalam tabel fisik dapat diakses dan dimanipulasi oleh pengguna melalui bahasa pemrograman atau aplikasi basis data.

Berikut adalah beberapa contoh penggunaan tabel fisik:

- Tabel "Akun" dapat digunakan untuk menyimpan data tentang akun, seperti nama, email, nomor telepon dan id akun.
- Tabel "Pelanggan" dapat digunakan untuk menyimpan data tentang pelanggan, seperti nama, alamat dan Id pelanggan.
- Tabel "Penjual" dapat digunakan untuk menyimpan data tentang penjual, seperti nama toko, alamat, dan Id penjual.

Tabel fisik memainkan peran penting dalam sistem basis data. Tabel fisik digunakan untuk menyimpan data yang diperlukan oleh sistem basis data. Dengan adanya tabel fisik, data dalam sistem basis data dapat diakses dan dimanipulasi dengan mudah.

F. Detail tambahan

Selain karakteristik yang telah disebutkan sebelumnya, tabel fisik juga memiliki beberapa karakteristik lain, yaitu:

- Primary key: Primary key adalah kolom atau kombinasi kolom yang unik dalam satu tabel. Primary key digunakan untuk mengidentifikasi baris dalam tabel.
- Foreign key: Foreign key adalah kolom atau kombinasi kolom yang merujuk ke primary key dari tabel lain. Foreign key digunakan untuk menghubungkan dua tabel.
- Index: Index adalah struktur data yang digunakan untuk mempercepat pencarian data dalam tabel.

Karakteristik-karakteristik tersebut dapat digunakan untuk meningkatkan performa dan efisiensi akses data dalam tabel fisik.

Berikut tabel fisik dari sistem E-commerce:

Tabel 1. Akun

Id_Akun	Email	Nama	No_telepon
11S22	charlos@gmail.com	Charlos	0812-1809-9848
12S22	yessi@gmail.com	Yessi	0812-6328-0007
13S22	esra@gmail.com	Esra	0858-3105-7136
14S22	kartika@gmail.com	Kartika	0813-6154-1794
15S22	christian@gmail.com	Christian	0812-6013-6699
16S22	chalvin@gmail.com	Chalvin	0853-6520-5722
17S22	anton@gmail.com	Anton	0822-6021-5832
18S22	lena@gmail.com	Lena	0812-7723-5987
19S22	dilan@gmail.com	Dilan	0813-8012-1714
20S22	daniel@gmail.com	Daniel	0853-9654-0004

Tabel 2. Pelanggan

Id_Pelanggan	Nama	Alamat
13S22	Chalvin	Medan
14S22	Yessi	Tarutung
16S22	Anton	Balige
11S22	Lena	Medan
12S22	Dilan	Sibolga

Tabel 3. Order

Id_Order	Nama Produk	Diskon	Harga	Kuantitas	Id_Pelanggan
11CP1	Baju	10%	Rp. 75.000	2	13S22
11CP2	Celana Pendek	20%	Rp. 70.000	3	14S22
11CP3	Sendal Gunung	15%	Rp. 180.000	1	16S22
11CP4	Sepatu	17%	Rp. 500.000	1	11S22
11CP5	Hoodie	10%	Rp. 250.000	1	12S22

Tabel 4. Pembayaran

ID_Pembayaran	Tanggal Pembayaran	Metode Pembayaran	Jumlah Pembayaran
23P01	1 Januari 2023	Internet Banking	2
23P02	1 Januari 2023	COD	3
23P03	1 Januari 2023	COD	1
23P04	1 Januari 2023	COD	1
23P05	1 Januari 2023	Internet Banking	1

Tabel 5. Detail Pesanan

No_Pesanan	Id_Order	Id_Produk	Nama Pesanan	Kuantitas	Tanggal Pesanan	Harga Satuan	Status Pesanan	Jasa Pengiriman
122	11CP1	11YS1	Baju	2	1 Januari 2023	Rp. 75.000	Sudah dibayar	JNT
222	11CP2	11YS2	Celana Pendek	3	1 Januari 2023	Rp. 70.000	Belum dibayar	JNT
322	11CP3	11YS3	Sendal Gunung	1	1 Januari 2023	Rp. 180.000	Belum dibayar	JNT
422	11CP4	11YS4	Sepatu	1	1 Januari 2023	Rp. 500.000	Belum dibayar	JNT
522	11CP5	11YS5	Hoodie	1	1 Januari 2023	Rp. 250.000	Sudah dibayar	JNT

Tabel 6. Penjual

ID_Penjual	Nama Toko	Alamat Toko
11S22	Charlos Shop	Jakarta Timur
13S22	Mari Belanja	Medan
14S22	Jamet Store	Kalimantan
15S22	Serba ada	Medan
20S22	Your Fashion	Jakarta Timur

Tabel 7. Pengiriman

No.Resi	Jasa Pengiriman	Tanggal Pesanan	Status Pengiriman	Alamat Pengiriman	ID_Penjual
JNT01	JNT	1 Januari 2023	Dikemas	Medan	11S22
JNT02	JNT	1 Januari 2023	Dalam Perjalanan	Medan	13S22
JNT03	JNT	1 Januari 2023	Dalam Perjalanan	Tarutung	14S22
JNT04	JNT	1 Januari 2023	Dikemas	Balige	15S22
JNT05	JNT	1 Januari 2023	Dikemas	Medan	20S22

Tabel 8. Produk

ID_Produk	Nama Produk	Harga	Kuantitas	Kategori	ID_Penjual	ID_Keranjang
11YS1	Baju	Rp. 75.000	2	Pakaian	11S22	P001
11YS2	Celana Pendek	Rp. 70.000	3	Pakaian	13S22	P002
11YS3	Sendal Gunung	Rp. 180.000	1	Sendal	14S22	P003
11YS4	Sepatu	Rp. 500.000	1	Sepatu	15S22	P004
11YS5	Hoodie	Rp. 250.000	1	Pakaian	20S22	P005

Tabel 9. Keranjang

ID_Keranjang	ID_Pelanggan	Kuantitas	Catatan
P001	13S22	2	Stok tersedia
P002	14S22	3	Stok habis
P003	16S22	1	Stok tersedia
P004	11S22	1	Stok habis
P005	12S22	1	stok tersedia

Tabel 10. Tinjauan

ID_Tinjauan	Isi_Tinjauan	Tanggal_Tinjauan	Rating	ID_Produk	ID_Pelanggan
001T1	Kualitas barang baik	6 Januari	4	11YS1	13S22
001T2	Kualitas barang sangat baik	7 Januari	5	11YS2	14S22
001T3	Kualitas barang baik	8 Januari	3	11YS3	16S22
001T4	Kualitas barang kurang baik	9 Januari	2	11YS4	11S22
001T5	Kualitas barang baik	10 Januari	4	11YS5	12S22

BAB 3

NORMALISASI

3.1 Normalisasi Bentuk Pertama (1NF)

Bentuk normal tahap pertama (1NF) terpenuhi jika sebuah tabel tidak memiliki atribut bernilai banyak (Multivalued Attribute) atau lebih dari satu atribut dengan domain nilai yang sama.

Tabel 11. Akun (1NF)

ID_Akun	Email	Nama	no_telepon	ID_Penjual	ID_Pelanggan

Tabel 12. Pelanggan (1NF)

ID_Pelanggan	Nama	Alamat

Tabel 13. Order (1NF)

ID_Order	Nama_Produk	Diskon	Harga	Kuantitas	ID_Pelanggan

Tabel 14. Pelanggan (1NF)

ID_Pembayaran	Tanggal Pembayaran	Metode Pembayaran	Jumlah Pembayaran

Tabel 15. Detail Pesanan (1NF)

No_Pesanan	ID_Order	ID_Produk	Nama Pesanan	Kuantitas	Tanggal Pesanan	Harga Satuan	Status Pesanan	Jasa Pengiriman

Tabel 16. Penjual (1NF)

ID_Penjual	Nama Toko	Alamat Toko

Tabel 17. Produk (1NF)

ID_Produk	Nama Produk	Harga	Kuantitas	Kategori	ID_Penjual	ID_Keranjang

....

Tabel 18. Keranjang (1NF)

ID_Keranjang	ID_Pelanggan	Kuantitas	Catatan

Tabel 19. Tinjauan (1NF)

ID_Tinjauan	Isi Tinjauan	Tanggal Tinjauan	Rating	ID_Produk	ID_Pelanggan

3.2 Normalisasi Bentuk Kedua (2NF)

Bentuk Normal tahap Kedua (2NF) terpenuhi jika pada sebuah tabel, semua atribut yang tidak termasuk dalam primary key memiliki ketergantungan fungsional (KF) pada primary key secara utuh. Sebuah tabel dikatakan tidak memenuhi 2NF, jika ketergantungannya hanya bersifat parsial (hanya tergantung pada sebagian dari primary key).

Tabel 20. Akun (2NF)

ID penjual	No.Telepon

ID_Pelanggan	No.Telepon

ID_Akun	Email	Nama	no_telepon

Tabel 21. Penjual (2NF)

ID_Pelanggan	nama

ID_Pelanggan	Alamat

Tabel 22. Order (2NF)

ID_Pelanggan

ID_Order	Nama produk	Diskon	Harga	Kuantitas

Tabel 23. Detail Pesanan (2NF)

Id_Order	No_Pesanan	Kuantitas	Harga satuan

Id_Produk	Nama_Pesanan	Tanggal Pesanan	Status Pesanan	Jasa Pengiriman

Tabel 24. Penjual (2NF)

ID_Penjual	Nama Toko

ID_Penjual	Alamat Toko

Tabel 25. Pengiriman (2NF)

No resi	Jasa Pengiriman	Tanggal Pengiriman

Tabel 26. Produk (2NF)

ID_Penjual	Nama produk	Kategori

ID_Keranjang	Kuantitas	Harga

Tabel 27. Keranjang (2NF)

ID_Keranjang	Kuantitas	Catatan

ID_Pelanggan	Kuantitas	Catatan

Tabel 28. Tinjauan (2NF)

ID_Tinjauan	Isi Tinjauan	Tanggal Tinjauan	ID_Produk

ID_Pelangan	Rating

3.3 Normalisasi Bentuk Ketiga (3NF)

3NF (Third Normal Form) adalah tingkat normalisasi dalam desain basis data yang menerapkan aturan tertentu untuk mengurangi redundansi data dan menjaga integritas data. Normalisasi adalah proses pengorganisasian struktur basis data untuk mengurangi redundansi dan ketergantungan data yang tidak diinginkan. Selain memenuhi aturan 2NF, tidak boleh ada ketergantungan transitif non-kunci. Ini berarti setiap kolom yang bukan bagian dari kunci utama tidak boleh bergantung pada kolom non-kunci lainnya. Penerapan 3NF membantu memastikan bahwa basis data terorganisir dengan baik, mengurangi redundansi, dan menghindari anomali penyimpanan yang mungkin terjadi saat mengelola data. Meskipun normalisasi memberikan struktur yang lebih baik, terkadang dapat meningkatkan kompleksitas query, terutama pada basis data yang besar, sehingga penerapan normalisasi harus seimbang dengan kebutuhan aplikasi dan performa sistem.

Tabel 29. Order (3NF)

ID_Pelanggan

ID_Order	Nama produk	Kuantitas

Diskon	Harga

Tabel 30. Pengiriman (3NF)

No resi	Jasa Pengiriman	Tanggal Pengiriman	Status Pengiriman	Alamat Pengiriman

No resi	Jasa Pengirimam	Tanggal Pengirimam

Jasa Pengiriman	Status Pengiriman	Alamat Pengiriman

Tabel 31. Produk (3NF)

ID_Penjual	Nama produk	Kategori

ID_Keranjang	Kuantitas	Harga

Nama produk	Kategori	Harga

BAB 4

IMPLEMENTASI

4.1 Implementasi Database ke SQL Server

SQL Server merupakan solusi yang diarahkan untuk menjalankan program-program terkait basis data. Sebagai Sistem Manajemen Basis Data (SMBD), SQL Server menyediakan serangkaian fungsi server yang mendukung manajemen dan akses data. SMBD umumnya, termasuk MySQL dan Microsoft SQL Server, sangat mengandalkan model klien-server untuk mengakses dan memanipulasi data dalam basis data mereka (Syahrial et al., 2018).

Model klien-server memungkinkan pengguna atau aplikasi klien untuk terhubung ke server basis data, yang bertanggung jawab untuk menyimpan dan mengelola data. Pada model ini, klien mengirim permintaan ke server untuk memanipulasi data atau melakukan operasi tertentu seperti penambahan, penghapusan, atau pengambilan informasi dari basis data. Kemudian, server memproses permintaan tersebut dan mengembalikan hasilnya ke klien. Ini menciptakan arsitektur yang memungkinkan akses aman dan terstruktur terhadap data.

4.2 Creating and Maintaining Database

Merancang database baru dan file yang digunakan untuk menyimpan database, membuat snapshot database, atau melampirkan database dari file terpisah dari database yang dibuat sebelumnya. Dalam hal ini, merancang database baru adalah proses awal dalam pengembangan aplikasi berbasis database. Proses ini menentukan struktur dan organisasi data dalam database. Selanjutnya, membuat snapshot database adalah proses membuat salinan statis dari database pada saat tertentu. Snapshot database dapat digunakan untuk tujuan backup, pemulihan data, atau pengujian, dan yang terakhir memasang database dari file terpisah adalah proses memasang database dari file terpisah. File terpisah ini dapat berupa file dump database atau file database yang telah di-export.

Pembuatan database dapat dituliskan dengan syntax Query SQL sebagai berikut :

Syntax

```
-- CREATING AND MAINTING DATABASE

CREATE DATABASE Proyek_Basdat_Kelompok01
ON PRIMARY (NAME = EcommercePrimary,
    FILENAME = 'C:\Users\asus\Documents\E-commerce.mdf',
    SIZE = 10MB,
    MAXSIZE = 10000MB,
    FILEGROWTH = 20%),
    (NAME = EcommerceSecondary,
    FILENAME = 'C:\Users\asus\Documents\E-commerce.ndf',
    SIZE = 5MB,
    MAXSIZE = 10000MB,
    FILEGROWTH = 30%)

LOG ON
    (NAME = EcommerceLog,
    FILENAME = 'C:\Users\asus\Documents\E-commerce.ldf',
    SIZE = 30MB,
    MAXSIZE = 5000MB,
    FILEGROWTH = 20%)

sp_helpdb Proyek_Basdat_Kelompok01
```

```
-- CREATING AND MAINTING DATABASE

CREATE DATABASE Proyek_Basdat_Kelompok01
ON PRIMARY (NAME = EcommercePrimary,
    FILENAME = 'C:\Users\asus\Documents\E-commerce.mdf',
    SIZE = 10MB,
    MAXSIZE = 10000MB,
    FILEGROWTH = 20%),
    (NAME = EcommerceSecondary,
    FILENAME = 'C:\Users\asus\Documents\E-commerce.ndf',
    SIZE = 5MB,
    MAXSIZE = 10000MB,
    FILEGROWTH = 30%)

LOG ON
    (NAME = EcommerceLog,
    FILENAME = 'C:\Users\asus\Documents\E-commerce.ldf',
    SIZE = 30MB,
    MAXSIZE = 5000MB,
    FILEGROWTH = 20%)

99 %  Messages
Commands completed successfully.

Completion time: 2023-11-30T00:10:36.6441281+07:00

99 %  ✓ Query executed successfully.
```

Gambar 4. Create Database Proyek_Basdat_Kelompok01

The screenshot shows the SQL Server Management Studio interface. In the top pane, a query window titled 'SQLQuery1.sql - ch..CHARLOS(asus (71))' contains the following T-SQL code:

```

CREATE DATABASE [Proyek_Basdat_Kelompok01]
    ON  PRIMARY
        (NAME = 'Ecommerce',  

        FILENAME = 'C:\Users\asus\Documents\E-commerce.ndf',  

        SIZE = 5MB,  

        MAXSIZE = 10000MB,  

        FILEGROWTH = 30%)  

    LOG ON  

        (NAME = EcommerceLog,  

        FILENAME = 'C:\Users\asus\Documents\E-commerce.ldf',  

        SIZE = 30MB,  

        MAXSIZE = 5000MB,  

        FILEGROWTH = 20%)  

sp_helpdb Proyek_Basdat_Kelompok01

```

In the bottom pane, the 'Messages' tab shows the results of the execution:

name	db_size	owner	dbid	created	status	compatibility_level
1 Proyek_Basdat_Kelompok01	16.00 MB	CHARLO(asus	11	Nov 30 2023	Status=ONLINE, Updateability=READ_WRITE, UserAcc...	150

Below this, another table shows file details:

name	fileid	filename	filegroup	size	maxsize	growth	usage
1 Proyek_Basdat_Kelompok01	1	C:\Program Files\Microsoft SQL Server\MSSQL15.MSS...	PRIMARY	8192 KB	Unlimited	65536 KB	data only
2 Proyek_Basdat_Kelompok01_log	2	C:\Program Files\Microsoft SQL Server\MSSQL15.MSS...	NULL	8192 KB	2147483648 KB	65536 KB	log only

A status bar at the bottom indicates: 'Query executed successfully.' and shows the session details: 'charlos (15.0 RTM) | CHARLO(asus (71) | Proyek_Basdat_Kelompok01 | 00:00:24 | 3 rows'.

Gambar 5. Hasil Ouput Create Database Ecommerce

Kode Program Create Database adalah sintak untuk melakukan pembuatan aturan pembuatan nama database dengan nama Proyek_Basdat_Kelompok01.

4.2.1 Creating, Insert and View Maintaining Tabel

Melakukan pembuatan beberapa Tabel beserta kolom-kolom, dan type data penyusunnya, serta perintah-perintah untuk menetapkan hubungan dan batasan-batasan data sistem E-commerce.

4.2.1.1 Creating Tabel Akun

Syntax

```
--Tabel Akun

CREATE TABLE Akun(
    Id_Akun varchar(10) PRIMARY KEY,
    Email varchar(50),
    Nama varchar(50),
    No_Telepon varchar(50),

);
```

The screenshot shows the SSMS interface with a query window titled 'SQLQuery1.sql - ch..CHARLOS\asus (71)*'. The query code is identical to the one above, creating the 'Akun' table and inserting two rows of data. The 'Messages' pane at the bottom indicates the command was completed successfully.

```
--Tabel Akun

CREATE TABLE Akun(
    Id_Akun varchar(10) PRIMARY KEY,
    Email varchar(50),
    Nama varchar(50),
    No_Telepon varchar(50),
);

select * from Akun

INSERT INTO Akun (Id_Akun, Email, Nama, No_Telepon)
VALUES
    ('11522', 'charlos@gmail.com', 'Carlos', '0812-1809-9848'),
    ('12522', 'yessi@gmail.com', 'Yessi', '0812-6328-0007'),
```

Messages

Commands completed successfully.
Completion time: 2023-11-30T01:38:49.816Z+07:00

99 %

99 %

Query executed successfully.

charlos (15.0 RTM) | CHARLOS\asus (71) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows

Gambar 6. Creating Tabel Akun

4.2.1.2 Viewing Tabel Akun

Syntax

```
select * from Akun
```

The screenshot shows a SQL Server Management Studio window titled "SQLQuery1.sql - ch...CHARLOS\asus (71)*". The code pane contains the following SQL statements:

```
--Tabel Akun
CREATE TABLE Akun(
    Id_Akun varchar(10) PRIMARY KEY,
    Email varchar(50),
    Nama varchar(50),
    No_Telepon varchar(50),
);
select * from Akun

INSERT INTO Akun (Id_Akun, Email, Nama, No_Telepon)
VALUES
    ('11S22', 'charlos@gmail.com', 'Charlos', '0812-1809-9848'),
    ('12S22', 'yessi@gmail.com', 'Yessi', '0812-6328-0007'),
    ('13S22', 'reza@gmail.com', 'Reza', '0812-3456-7890')
```

The results pane shows the following table structure:

Id_Akun	Email	Nama	No_Telepon
11S22	charlos@gmail.com	Charlos	0812-1809-9848
12S22	yessi@gmail.com	Yessi	0812-6328-0007
13S22	reza@gmail.com	Reza	0812-3456-7890

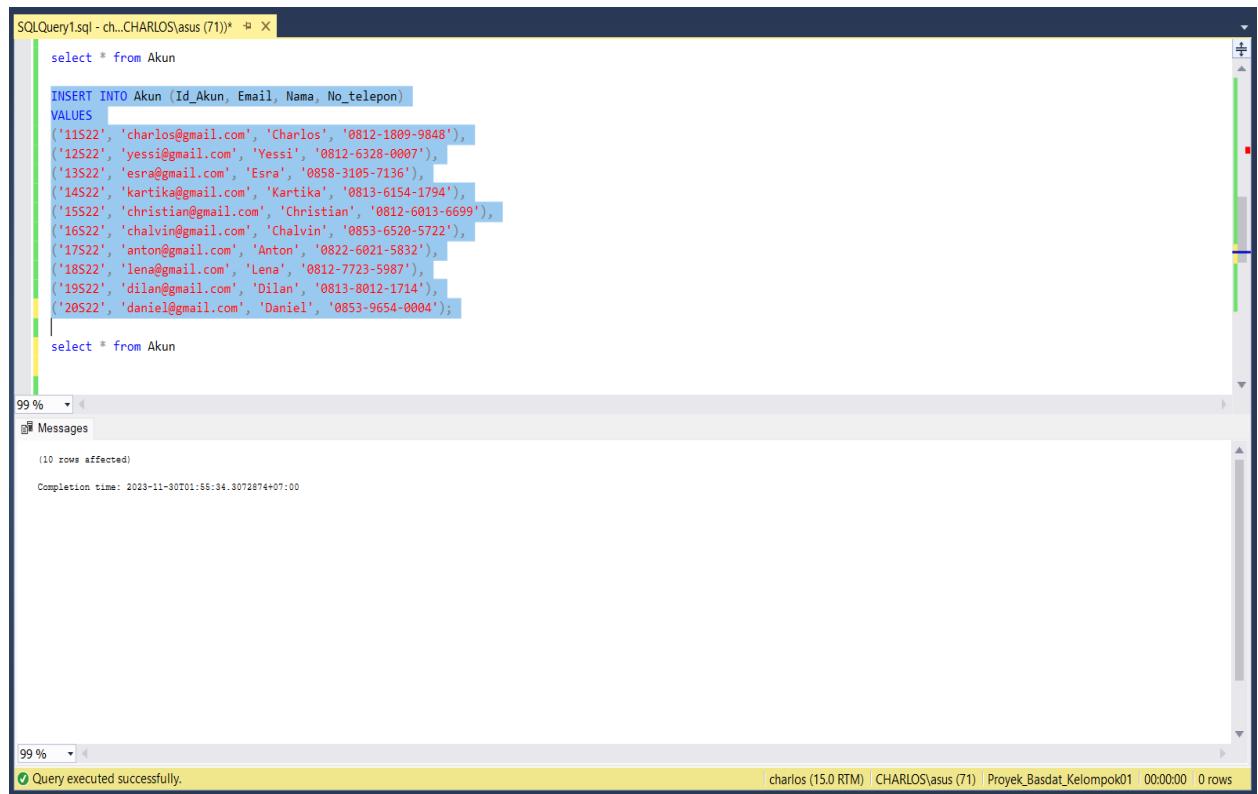
At the bottom, a message bar indicates: "Query executed successfully." and shows connection details: charlos (15.0 RTM) | CHARLOS\asus (71) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows.

Gambar 7. Hasil Output Creating Tabel Akun

4.2.1.3 Inserting Value Tabel Akun

Syntax

```
INSERT INTO Akun (Id_Akun, Email, Nama, No_telepon)
VALUES
('11S22', 'charlos@gmail.com', 'Carlos', '0812-1809-9848'),
('12S22', 'yessi@gmail.com', 'Yessi', '0812-6328-0007'),
('13S22', 'esra@gmail.com', 'Esra', '0858-3105-7136'),
('14S22', 'kartika@gmail.com', 'Kartika', '0813-6154-1794'),
('15S22', 'christian@gmail.com', 'Christian', '0812-6013-6699'),
('16S22', 'chalvin@gmail.com', 'Chalvin', '0853-6520-5722'),
('17S22', 'anton@gmail.com', 'Anton', '0822-6021-5832'),
('18S22', 'lena@gmail.com', 'Lena', '0812-7723-5987'),
('19S22', 'dilan@gmail.com', 'Dilan', '0813-8012-1714'),
('20S22', 'daniel@gmail.com', 'Daniel', '0853-9654-0004');
```



The screenshot shows the SSMS interface with a query window titled "SQLQuery1.sql - ch...CHARLOS\asus (71)*". The query is:

```
select * from Akun
INSERT INTO Akun (Id_Akun, Email, Nama, No_telepon)
VALUES
('11S22', 'charlos@gmail.com', 'Carlos', '0812-1809-9848'),
('12S22', 'yessi@gmail.com', 'Yessi', '0812-6328-0007'),
('13S22', 'esra@gmail.com', 'Esra', '0858-3105-7136'),
('14S22', 'kartika@gmail.com', 'Kartika', '0813-6154-1794'),
('15S22', 'christian@gmail.com', 'Christian', '0812-6013-6699'),
('16S22', 'chalvin@gmail.com', 'Chalvin', '0853-6520-5722'),
('17S22', 'anton@gmail.com', 'Anton', '0822-6021-5832'),
('18S22', 'lena@gmail.com', 'Lena', '0812-7723-5987'),
('19S22', 'dilan@gmail.com', 'Dilan', '0813-8012-1714'),
('20S22', 'daniel@gmail.com', 'Daniel', '0853-9654-0004');
select * from Akun
```

The status bar at the bottom indicates "99 %", "Messages", "(10 rows affected)", "Completion time: 2023-11-30T01:55:34.3072874+07:00", and "charlos (15.0 RTM) | CHARLOS\asus (71) | Projek_Basdat_Kelompok01 | 00:00:00 | 0 rows". A green message bar at the bottom says "Query executed successfully."

Gambar 8. Insert Value into Tabel Akun

Syntax

```
select * from Akun
```

The screenshot shows the SQL Query window of SQL Server Management Studio. The code entered is:

```
select * from Akun

INSERT INTO Akun (Id_Akun, Email, Nama, No_telepon)
VALUES
('11S22', 'charlos@gmail.com', 'Carlos', '0812-1809-9848'),
('12S22', 'yessi@gmail.com', 'Yessi', '0812-6328-0007'),
('13S22', 'esra@gmail.com', 'Esra', '0858-3105-7136'),
('14S22', 'kartika@gmail.com', 'Kartika', '0813-6154-1794'),
('15S22', 'christian@gmail.com', 'Christian', '0812-6013-6699'),
('16S22', 'chalvin@gmail.com', 'Chalvin', '0853-6520-5722'),
('17S22', 'anton@gmail.com', 'Anton', '0822-6021-5832'),
('18S22', 'lena@gmail.com', 'Lena', '0812-7723-5987'),
('19S22', 'dilan@gmail.com', 'Dilan', '0813-8012-1714'),
('20S22', 'daniel@gmail.com', 'Daniel', '0853-9654-0004');

select * from Akun
```

The results pane displays the data inserted into the Akun table:

Id_Akun	Email	Nama	No_Telepon	
1	11S22	charlos@gmail.com	Charlo	0812-1809-9848
2	12S22	yessi@gmail.com	Yessi	0812-6328-0007
3	13S22	esra@gmail.com	Esra	0858-3105-7136
4	14S22	kartika@gmail.com	Kartika	0813-6154-1794
5	15S22	christian@gmail.com	Christian	0812-6013-6699
6	16S22	chalvin@gmail.com	Chalvin	0853-6520-5722
7	17S22	anton@gmail.com	Anton	0822-6021-5832
8	18S22	lena@gmail.com	Lena	0812-7723-5987
9	19S22	dilan@gmail.com	Dilan	0813-8012-1714
10	20S22	daniel@gmail.com	Daniel	0853-9654-0004

At the bottom of the results pane, a message indicates: "Query executed successfully." To the right, status information is shown: charlos (15.0 RTM) | CHARLOS\asus (71) | Projek_Basdat_Kelompok01 | 00:00:00 | 10 rows.

Gambar 9. Hasil Output Insert Value into Tabel Akun

4.2.1.4 Creating Tabel Pelanggan

Syntax

```
CREATE TABLE Pelanggan (
    Id_Pelanggan varchar(10) PRIMARY KEY,
    Nama varchar(50),
    Alamat varchar(50),
);
```

The screenshot shows the SSMS interface with a query window containing the following SQL code:

```
-- Tabel Pelanggan
CREATE TABLE Pelanggan (
    Id_Pelanggan varchar(10) PRIMARY KEY,
    Nama varchar(50),
    Alamat varchar(50),
);
select * from Pelanggan
INSERT INTO NamaLabel (Id_Pelanggan, Nama, Alamat)
VALUES
('16522', 'Chalvin', 'Medan'),
('12522', 'Yessi', 'Tarutung'),
('17522', 'Anton', 'Balige'),
('18522', 'Lona', 'Medan')
```

The status bar at the bottom indicates "Query executed successfully." and "0 rows".

Gambar 10. Creating Tabel Pelanggan

4.2.1.5 Viewing Tabel Pelanggan

Syntax

```
select * from Pelanggan
```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS(asus (60))". The code is as follows:

```
-- Tabel Pelanggan
CREATE TABLE Pelanggan (
    Id_Pelanggan varchar(10) PRIMARY KEY,
    Nama varchar(50),
    Alamat varchar(50),
);
select * from Pelanggan

INSERT INTO NamaTabel (Id_Pelanggan, Nama, Alamat)
VALUES
('16522', 'Chalvin', 'Medan'),
('12522', 'Yessi', 'Tarutung'),
('17522', 'Anton', 'Bulige'),
('18522', 'Lena', 'Medan'),
('19522', 'Nita', 'Sukabumi').
```

The status bar at the bottom indicates "99 %". Below the code, there are two tabs: "Results" and "Messages". The "Results" tab is selected and shows a table structure with columns "Id_Pelanggan", "Nama", and "Alamat". The "Messages" tab shows a success message: "Query executed successfully." The status bar at the bottom right shows "charlos (15.0 RTM) | CHARLOS(asus (60)) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows".

Gambar 11. Hasil Output Creating Tabel Pelanggan

4.2.1.6 Inserting Tabel Pelanggan

Syntax

```
INSERT INTO Pelanggan (Id_Pelanggan, Nama, Alamat)
VALUES
('13S22', 'Chalvin', 'Medan'),
('14S22', 'Yessi', 'Tarutung'),
('16S22', 'Anton', 'Balige'),
('11S22', 'Lena', 'Medan'),
('12S22', 'Dilan', 'Sibolga');
```

```
SQLQuery1.sql - ch...CHARLOS\asus (60) ×
select * from Pelanggan
INSERT INTO Pelanggan (Id_Pelanggan, Nama, Alamat)
VALUES
('16S22', 'Chalvin', 'Medan'),
('12S22', 'Yessi', 'Tarutung'),
('17S22', 'Anton', 'Balige'),
('18S22', 'Lena', 'Medan'),
('19S22', 'Dilan', 'Sibolga');

select * from Pelanggan

99 % ▾
Messages
(5 rows affected)
Completion time: 2023-11-30T19:01:29.1945020+07:00

99 % ▾
Query executed successfully. charlos (15.0 RTM) | CHARLOS\asus (60) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows
```

Gambar 12. Insert Value into Tabel Pelanggan

Syntax

```
select * from Pelanggan
```

```

SQLQuery1.sql - ch...CHARLOS\asus (58)* ×
CREATE TABLE Pelanggan (
    Id_Pelanggan varchar(10) PRIMARY KEY,
    Nama varchar(50),
    Alamat varchar(50),
);
select * from Pelanggan

=INSERT INTO Pelanggan (Id_Pelanggan, Nama, Alamat)
VALUES
('13522', 'Chalvin', 'Medan'),
('14522', 'Yessi', 'Tarutung'),
('16522', 'Anton', 'Balige'),
('11522', 'Lena', 'Medan'),
('12522', 'Dilan', 'Sibolga');

select * from Pelanggan

```

Results Messages

	Id_Pelanggan	Nama	Alamat
1	11522	Lena	Medan
2	12522	Dilan	Sibolga
3	13522	Chalvin	Medan
4	14522	Yessi	Tarutung
5	16522	Anton	Balige

Query executed successfully. | charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 5 rows

Gambar 13. Hasil Output Insert Value into Tabel Pelanggan

4.2.1.7 Creating Tabel Order

Syntax

```

CREATE TABLE Orders (
    Id_Order varchar(10) PRIMARY KEY,
    Nama_Produk varchar(50),
    Diskon varchar(10),
    Harga varchar(10),
    Kuantitas varchar(10),
    Id_Pelanggan varchar(10) REFERENCES Pelanggan (Id_Pelanggan)
);

```

SQLQuery1.sql - ch...CHARLOS\asus (60)*

```
-- Tabel Order

CREATE TABLE Orders (
    Id_Order varchar(10) PRIMARY KEY,
    Nama_Produk varchar(50),
    Diskon varchar(10),
    Harga varchar(10),
    Kuantitas varchar(10),
    Id_Pelanggan varchar(10) REFERENCES Pelanggan (Id_Pelanggan)
);

select * from Orders
```

99 %

Messages

Commands completed successfully.

Completion time: 2023-11-30T19:21:02.4619601+07:00

99 %

Query executed successfully.

Gambar 14. Creating Tabel Orders

4.2.1.8 Viewing Tabel Order

Syntax

```
select * from Orders
```

The screenshot shows a SQL query window titled 'SQLQuery1.sql - ch...CHARLOS(asus (60))'. The code creates a table 'Orders' with columns: Id_Order (primary key), Nama_Produk, Diskon, Harga, Kuantitas, and Id_Pelanggan. A select query is also present. Below the code, the results pane shows a table structure with columns: Id_Order, Nama_Produk, Diskon, Harga, Kuantitas, and Id_Pelanggan. At the bottom, a message indicates the query was executed successfully.

```
-- Tabel Order
CREATE TABLE Orders (
    Id_Order varchar(10) PRIMARY KEY,
    Nama_Produk varchar(50),
    Diskon varchar(10),
    Harga varchar(10),
    Kuantitas varchar(10),
    Id_Pelanggan varchar(10) REFERENCES Pelanggan (Id_Pelanggan)
);
select * from Orders
```

99 % Results Messages

Id_Order	Nama_Produk	Diskon	Harga	Kuantitas	Id_Pelanggan

Query executed successfully. | charlos (15.0 RTM) | CHARLOS(asus (60)) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows

Gambar 15. Hasil Output Creating Tabel Orders

4.2.1.9 Inserting Tabel Order

Syntax

```
INSERT INTO Orders (Id_Order, Nama_Produk, Diskon, Harga, Kuantitas, Id_Pelanggan)
VALUES
('11CP1', 'Baju', '10%', 'Rp.75.000', '2', '13S22'),
('11CP2', 'Celana Pendek', '20%', 'Rp.70.000', '3', '14S22'),
('11CP3', 'Sendal Gunung', '15%', 'Rp.180.000', '1', '16S22'),
('11CP4', 'Sepatu', '17%', 'Rp.500.000', '1', '11S22'),
('11CP5', 'Hoodie', '10%', 'Rp.250.000', '1', '12S22');
```

```

SQLQuery1.sql - ch...CHARLOS\asus (58)  ✎ ×
Harga varchar(10),
Kuantitas varchar(10),
Id_Pelanggan varchar(10) REFERENCES Pelanggan (Id_Pelanggan)
);
select * from Orders
--INSERT INTO Orders (Id_Order, Nama_Produk, Diskon, Harga, Kuantitas, Id_Pelanggan)
VALUES
('11CP1', 'Baju', '10%', 'Rp.75.000', '2', '13522'),
('11CP2', 'Celana Pendek', '20%', 'Rp.70.000', '3', '14522'),
('11CP3', 'Sendal Gunung', '15%', 'Rp.180.000', '1', '16522'),
('11CP4', 'Sepatu', '17%', 'Rp.500.000', '1', '11522'),
('11CP5', 'Hoodie', '10%', 'Rp.250.000', '1', '12522');

select * from Orders

-- Tabel Pembayaran

CREATE TABLE Pembayaran (
ID_Pembayaran varchar(10) PRIMARY KEY,
Tanggal_Pembayaran date,
);

```

99 %

Messages

(6 rows affected)

Completion time: 2023-12-01T00:59:10.9059953+07:00

99 %

Query executed successfully.

Gambar 16. Insert Value into Tabel Orders

Syntax

```
select * from Orders
```

```

SQLQuery1.sql - ch...CHARLOS\asus (58)  ✎ ×
C:\Users\asus\Documents\E-commerce\SQLQuery1.sql - charlos.Proyek_Basdat_Kelompok01 (CHARLOS\asus (58))
Id_Pelanggan varchar(10) REFERENCES Pelanggan (Id_Pelanggan)
);
select * from Orders
--INSERT INTO Orders (Id_Order, Nama_Produk, Diskon, Harga, Kuantitas, Id_Pelanggan)
VALUES
('11CP1', 'Baju', '10%', 'Rp.75.000', '2', '13522'),
('11CP2', 'Celana Pendek', '20%', 'Rp.70.000', '3', '14522'),
('11CP3', 'Sendal Gunung', '15%', 'Rp.180.000', '1', '16522'),
('11CP4', 'Sepatu', '17%', 'Rp.500.000', '1', '11522'),
('11CP5', 'Hoodie', '10%', 'Rp.250.000', '1', '12522');

select * from Orders

-- Tabel Pembayaran

CREATE TABLE Pembayaran (
ID_Pembayaran varchar(10) PRIMARY KEY,
Tanggal_Pembayaran date,
);

```

99 %

Results

	Id_Order	Nama_Produk	Diskon	Harga	Kuantitas	Id_Pelanggan
1	11CP1	Baju	10%	Rp.75.000	2	13522
2	11CP2	Celana Pendek	20%	Rp.70.000	3	14522
3	11CP3	Sendal Gunung	15%	Rp.180.000	1	16522
4	11CP4	Sepatu	17%	Rp.500.000	1	11522
5	11CP5	Hoodie	10%	Rp.250.000	1	12522

99 %

Messages

Query executed successfully.

Gambar 17. Hasil Output Insert Value into Tabel Orders

4.2.1.10 Creating Tabel Pembayaran

Syntax

```
CREATE TABLE Pembayaran (
    ID_Pembayaran varchar(10) PRIMARY KEY,
    Tanggal_Pembayaran date,
    Metode_Pembayaran varchar(50),
    Jumlah_Pembayaran int
);
```

The screenshot shows a SQL query window in SSMS with the following content:

```
-- Tabel Pembayaran
CREATE TABLE Pembayaran (
    ID_Pembayaran varchar(10) PRIMARY KEY,
    Tanggal_Pembayaran date,
    Metode_Pembayaran varchar(50),
    Jumlah_Pembayaran int
);
select * from Pembayaran

--INSERT INTO Pembayaran (ID_Pembayaran, Tanggal_Pembayaran, Metode_Pembayaran, Jumlah_Pembayaran)
VALUES
('23P01', '2023-01-01', 'Internet Banking', 2),
('23P02', '2023-01-02', 'COD', 3),
('23P03', '2023-01-03', 'COD', 1),
('23P04', '2023-01-04', 'COD', 1),
('23P05', '2023-01-05', 'Internet Banking', 1);
select * from Pembayaran
```

The Messages pane at the bottom shows:

- Commands completed successfully.
- Completion time: 2023-11-30T20:25:03.0794236+07:00

The status bar at the bottom right indicates: charlos (15.0 RTM) | CHARLOS\asus (58) | master | 00:00:00 | 0 rows

Gambar 18. Creating Tabel Pembayaran

4.2.1.11 Viewing Tabel Pembayaran

Syntax

```
select * from Pembayaran
```

The screenshot shows a SQL query window in SSMS. The code creates a table 'Pembayaran' with columns ID_Pembayaran, Tanggal_Pembayaran, Metode_Pembayaran, and Jumlah_Pembayaran. It then inserts five rows of data into the table and finally selects all columns from the table.

```
-- Tabel Pembayaran
CREATE TABLE Pembayaran (
    ID_Pembayaran varchar(10) PRIMARY KEY,
    Tanggal_Pembayaran date,
    Metode_Pembayaran varchar(50),
    Jumlah_Pembayaran int
);
select * from Pembayaran
INSERT INTO Pembayaran (ID_Pembayaran, Tanggal_Pembayaran, Metode_Pembayaran, Jumlah_Pembayaran)
VALUES
('23P01', '2023-01-01', 'Internet Banking', 2),
('23P02', '2023-01-02', 'COD', 3),
('23P03', '2023-01-03', 'COD', 1),
('23P04', '2023-01-04', 'COD', 1),
('23P05', '2023-01-05', 'Internet Banking', 1);
select * from Pembayaran
```

Results Messages

ID_Pembayaran	Tanggal_Pembayaran	Metode_Pembayaran	Jumlah_Pembayaran
23P01	2023-01-01	Internet Banking	2
23P02	2023-01-02	COD	3
23P03	2023-01-03	COD	1
23P04	2023-01-04	COD	1
23P05	2023-01-05	Internet Banking	1

Query executed successfully. charlos (15.0 RTM) | CHARLOS\asus (58) | master | 00:00:00 | 0 rows

Gambar 19. Hasil Output creating Tabel Pembayaran

4.2.1.12 Inserting Tabel Pembayaran

Syntax

```
INSERT INTO Pembayaran (ID_Pembayaran, Tanggal_Pembayaran, Metode_Pembayaran,
Jumlah_Pembayaran)
VALUES
('23P01', '2023-01-01', 'Internet Banking', 2),
('23P02', '2023-01-02', 'COD', 3),
('23P03', '2023-01-03', 'COD', 1),
('23P04', '2023-01-04', 'COD', 1),
('23P05', '2023-01-05', 'Internet Banking', 1);
```

```

-- Tabel Pembayaran

CREATE TABLE Pembayaran (
    ID_Pembayaran varchar(10) PRIMARY KEY,
    Tanggal_Pembayaran date,
    Metode_Pembayaran varchar(50),
    Jumlah_Pembayaran int
);

select * from Pembayaran

INSERT INTO Pembayaran (ID_Pembayaran, Tanggal_Pembayaran, Metode_Pembayaran, Jumlah_Pembayaran)
VALUES
    ('23P01', '2023-01-01', 'Internet Banking', 2),
    ('23P02', '2023-01-02', 'COD', 3),
    ('23P03', '2023-01-03', 'COD', 1),
    ('23P04', '2023-01-04', 'COD', 1),
    ('23P05', '2023-01-05', 'Internet Banking', 1);

select * from Pembayaran

```

99 % ▾

Messages

(5 rows affected)

Completion time: 2023-11-30T20:54:22.1226882+07:00

99 % ▾

Query executed successfully.

charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows

Gambar 20. Insert Value into Tabel Pembayaran

Syntax

```
select * from Pembayaran
```

```

-- Tabel Pembayaran

CREATE TABLE Pembayaran (
    ID_Pembayaran varchar(10) PRIMARY KEY,
    Tanggal_Pembayaran date,
    Metode_Pembayaran varchar(50),
    Jumlah_Pembayaran int
);

select * from Pembayaran

INSERT INTO Pembayaran (ID_Pembayaran, Tanggal_Pembayaran, Metode_Pembayaran, Jumlah_Pembayaran)
VALUES
    ('23P01', '2023-01-01', 'Internet Banking', 2),
    ('23P02', '2023-01-02', 'COD', 3),
    ('23P03', '2023-01-03', 'COD', 1),
    ('23P04', '2023-01-04', 'COD', 1),
    ('23P05', '2023-01-05', 'Internet Banking', 1);

select * from Pembayaran

```

99 % ▾

Results

ID_Pembayaran	Tanggal_Pembayaran	Metode_Pembayaran	Jumlah_Pembayaran
23P01	2023-01-01	Internet Banking	2
23P02	2023-01-02	COD	3
23P03	2023-01-03	COD	1
23P04	2023-01-04	COD	1
23P05	2023-01-05	Internet Banking	1

99 % ▾

Messages

Query executed successfully.

charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 5 rows

Gambar 21. Hasil Output Insert Value into Tabel Pembayaran

4.2.1.13 Creating Tabel Detail Pesanan

Syntax

```
CREATE TABLE Detail_Pesanan (
    No_Pesanan INT PRIMARY KEY,
    Id_Order VARCHAR(10),
    Id_Produk VARCHAR(10),
    Nama_Pesanan VARCHAR(50),
    Kuantitas INT,
    Tanggal_Pesanan DATE,
    Harga_Satuan VARCHAR(20),
    Status_Pesanan VARCHAR(50),
    Jasa_Pengiriman VARCHAR(50),
    CONSTRAINT FK_Id_Order FOREIGN KEY (Id_Order) REFERENCES Orders(Id_Order)
);
```

The screenshot shows the SSMS interface with a query window titled "SQLQuery1.sql - ch...CHARLOS(asus (58))". The code in the window is as follows:

```
-- Tabel Detail Pesanan
CREATE TABLE Detail_Pesanan (
    No_Pesanan INT PRIMARY KEY,
    Id_Order VARCHAR(10),
    Id_Produk VARCHAR(10),
    Nama_Pesanan VARCHAR(50),
    Kuantitas INT,
    Tanggal_Pesanan DATE,
    Harga_Satuan VARCHAR(20),
    Status_Pesanan VARCHAR(50),
    Jasa_Pengiriman VARCHAR(50)
    CONSTRAINT FK_Id_Order FOREIGN KEY (Id_Order) REFERENCES Orders(Id_Order)
);
select * from Detail_Pesanan
INSERT INTO Detail_Pesanan (No_Pesanan, Id_Order, Id_Produk, Nama_Pesanan, Kuantitas, Tanggal_Pesanan, Harga_Satuan, Status_Pesanan, Jasa_Pengiriman)
VALUES
(1, '11CP1', '11YS1', 'Baju', 2, NULL, 'Rp. 75.000', 'Sudah dibayar', 'JNT'),
(2, '11CP2', '11YS2', 'Celana Pendek', 3, NULL, 'Rp. 70.000', 'Belum dibayar', 'JNT'),
(3, '11CP3', '11YS3', 'Sendal Gunung', 1, NULL, 'Rp. 180.000', 'Belum dibayar', 'JNT'),
(4, '11CP4', '11YS4', 'Sepatu', 1, NULL, 'Rp. 500.000', 'Belum dibayar', 'JNT'),
```

The status bar at the bottom indicates "Messages", "Commands completed successfully.", and "Completion time: 2023-11-30T20:40:10.7833194+07:00". The status bar also shows "charlos (15.0 RTM) | CHARLOS(asus (58)) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows".

Gambar 22. Creating Tabel Detail Pesanan

4.2.1.14 Viewing Tabel Detail Pesanan

Syntax

```
select * from Detail_Pesanan
```

The screenshot shows the SQL Query Editor in SQL Server Management Studio. The code in the editor is:

```
-- Tabel Detail Pesanan
CREATE TABLE Detail_Pesanan (
    No_Pesanan INT PRIMARY KEY,
    Id_Order VARCHAR(10),
    Id_Produk VARCHAR(10),
    Nama_Pesanan VARCHAR(50),
    Kuantitas INT,
    Tanggal_Pesanan DATE,
    Harga_Satuan VARCHAR(20),
    Status_Pesanan VARCHAR(50),
    Jasa_Pengiriman VARCHAR(50),
    CONSTRAINT FK_Id_Order FOREIGN KEY (Id_Order) REFERENCES Orders(Id_Order)
);
select * from Detail_Pesanan

INSERT INTO Detail_Pesanan (No_Pesanan, Id_Order, Id_Produk, Nama_Pesanan, Kuantitas, Tanggal_Pesanan, Harga_Satuan, Status_Pesanan, Jasa_Pengiriman)
VALUES
(1, '11CP1', '11YS1', 'Baju', 2, NULL, 'Rp. 75.000', 'Sudah dibayar', 'JNT'),
(2, '11CP2', '11YS2', 'Celana Pendek', 3, NULL, 'Rp. 70.000', 'Belum dibayar', 'JNT'),
(3, '11CP3', '11YS3', 'Sendal Gunung', 1, NULL, 'Rp. 180.000', 'Belum dibayar', 'JNT'),
(4, '11CP4', '11YS4', 'Sepatu', 1, NULL, 'Rp. 500.000', 'Belum dibayar', 'JNT'),
```

The results pane is empty, indicating no rows were returned. The status bar at the bottom right shows "charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows".

Gambar 23. Hasil Output Creating Tabel Detail Pesanan

4.2.1.15 Inserting Tabel Detail Pesanan

Syntax

```
INSERT INTO Detail_Pesanan (No_Pesanan, Id_Order, Id_Produk, Nama_Pesanan, Kuantitas,
Tanggal_Pesanan, Harga_Satuan, Status_Pesanan, Jasa_Pengiriman)
VALUES
(122, '11CP1', '11YS1', 'Baju', 2, '2023-01-01', 'Rp. 75.000', 'Sudah dibayar', 'JNT'),
(222, '11CP2', '11YS2', 'Celana Pendek', 3, '2023-01-01', 'Rp. 70.000', 'Belum
dibayar', 'JNT'),
(322, '11CP3', '11YS3', 'Sendal Gunung', 1, '2023-01-01', 'Rp. 180.000', 'Belum
dibayar', 'JNT'),
(422, '11CP4', '11YS4', 'Sepatu', 1, '2023-01-01', 'Rp. 500.000', 'Belum dibayar',
'JNT'),
(522, '11CP5', '11YS5', 'Hoodie', 1, '2023-01-01', 'Rp. 250.000', 'Sudah dibayar',
'JNT');
```

```

SQLQuery1.sql - ch...CHARLOS\asus (58)  ↗ ×
    Status_Pesanan VARCHAR(50),
    Jasa_Pengiriman VARCHAR(50),
    CONSTRAINT FK_Id_Order FOREIGN KEY (Id_Order) REFERENCES Orders(Id_Order)
);
select * from Detail_Pesanan

INSERT INTO Detail_Pesanan (No_Pesanan, Id_Order, Id_Produk, Nama_Pesanan, Kuantitas, Tanggal_Pesanan, Harga_Satuan, Status_Pesanan, Jasa_Pengiriman)
VALUES
(122, '11CP1', '11YS1', 'Baju', 2, '2023-01-01', 'Rp. 75.000', 'Sudah dibayar', 'JNT'),
(222, '11CP2', '11YS2', 'Celana Pendek', 3, '2023-01-01', 'Rp. 70.000', 'Belum dibayar', 'JNT'),
(322, '11CP3', '11YS3', 'Sendal Gunung', 1, '2023-01-01', 'Rp. 180.000', 'Belum dibayar', 'JNT'),
(422, '11CP4', '11YS4', 'Sepatu', 1, '2023-01-01', 'Rp. 500.000', 'Belum dibayar', 'JNT'),
(522, '11CP5', '11YS5', 'Hoodie', 1, '2023-01-01', 'Rp. 250.000', 'Sudah dibayar', 'JNT');

select * from Detail_Pesanan

```

99 % ↴

Messages

(5 rows affected)

Completion time: 2023-11-30T20:53:04.0919820+07:00

99 % ↴

Query executed successfully.

charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows

Gambar 24. Insert Value into Tabel Detail Pesanan

Syntax

```
select * from Detail_Pesanan
```

```

SQLQuery1.sql - ch...CHARLOS\asus (58)  ↗ ×
    Kuantitas INT,
    Tanggal_Pesanan DATE,
    Harga_Satuan VARCHAR(20),
    Status_Pesanan VARCHAR(50),
    Jasa_Pengiriman VARCHAR(50),
    CONSTRAINT FK_Id_Order FOREIGN KEY (Id_Order) REFERENCES Orders(Id_Order)
);
select * from Detail_Pesanan

SELECT
    No_Pesanan, Id_Order, Id_Produk, Nama_Pesanan, Kuantitas, Tanggal_Pesanan, Harga_Satuan, Status_Pesanan, Jasa_Pengiriman
FROM
    Detail_Pesanan

```

99 % ↴

Results

No_Pesanan	Id_Order	Id_Produk	Nama_Pesanan	Kuantitas	Tanggal_Pesanan	Harga_Satuan	Status_Pesanan	Jasa_Pengiriman
1 122	11CP1	11YS1	Baju	2	2023-01-01	Rp. 75.000	Sudah dibayar	JNT
2 222	11CP2	11YS2	Celana Pendek	3	2023-01-01	Rp. 70.000	Belum dibayar	JNT
3 322	11CP3	11YS3	Sendal Gunung	1	2023-01-01	Rp. 180.000	Belum dibayar	JNT
4 422	11CP4	11YS4	Sepatu	1	2023-01-01	Rp. 500.000	Belum dibayar	JNT
5 522	11CP5	11YS5	Hoodie	1	2023-01-01	Rp. 250.000	Sudah dibayar	JNT

99 % ↴

Messages

Query executed successfully.

charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 5 rows

Gambar 25. Hasil Output Insert Value into Tabel Detail Pesanan

4.2.1.16 Creating Tabel Penjual

Syntax

```
CREATE TABLE Penjual (
    ID_Penjual VARCHAR(10) PRIMARY KEY,
    Nama_Toko VARCHAR(50),
    Alamat_Toko VARCHAR(100)
);
```

The screenshot shows the SSMS interface with a query window titled 'SQLQuery1.sql - ch...CHARLOS(asus (58))'. The code entered is:

```
-- Tabel Penjual
CREATE TABLE Penjual (
    ID_Penjual VARCHAR(10) PRIMARY KEY,
    Nama_Toko VARCHAR(50),
    Alamat_Toko VARCHAR(100)
);
select * from Penjual
INSERT INTO Penjual (ID_Penjual, Nama_Toko, Alamat_Toko)
VALUES
('11522', 'Carlos Shop', 'Jakarta Timur'),
('13522', 'Mari Belanja', 'Medan'),
('14522', 'Jamat Store', 'Kalimantan'),
('15522', 'Serba ada', 'Medan'),
('20522', 'Your Fashion', 'Jakarta Timur');
select * from Penjual
```

The status bar at the bottom indicates '99 %' completion, 'Messages' with 'Commands completed successfully.', 'Completion time: 2023-11-30T20:57:33.8027345+07:00', and a green checkmark icon followed by 'Query executed successfully.'

Gambar 26. Creating Tabel Penjual

4.2.1.17 Viewing Tabel Penjual

Syntax

```
select * from Penjual
```

The screenshot shows the SQL Query1.sql - ch...CHARLOS(asus (58)) window in SQL Server Management Studio. The query window contains the following SQL code:

```
-- Tabel Penjual
CREATE TABLE Penjual (
    ID_Penjual VARCHAR(10) PRIMARY KEY,
    Nama_Toko VARCHAR(50),
    Alamat_Toko VARCHAR(100)
);
select * from Penjual
INSERT INTO Penjual (ID_Penjual, Nama_Toko, Alamat_Toko)
VALUES
('11S22', 'Carlos Shop', 'Jakarta Timur'),
('13S22', 'Mari Belanja', 'Medan'),
('14S22', 'Jemet Store', 'Kalimantan'),
('15S22', 'Serba ada', 'Medan'),
('20S22', 'Your Fashion', 'Jakarta Timur');
select * from Penjual
```

The results pane below shows the table structure and the inserted data:

ID_Penjual	Nama_Toko	Alamat_Toko
11S22	Carlos Shop	Jakarta Timur
13S22	Mari Belanja	Medan
14S22	Jemet Store	Kalimantan
15S22	Serba ada	Medan
20S22	Your Fashion	Jakarta Timur

At the bottom, a message indicates: "Query executed successfully." and "charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows".

Gambar 27. Hasil Output Creating Tabel Penjual

4.2.1.18 inserting Tabel Penjual

Syntax

```
INSERT INTO Penjual (ID_Penjual, Nama_Toko, Alamat_Toko)
VALUES
('11S22', 'Carlos Shop', 'Jakarta Timur'),
('13S22', 'Mari Belanja', 'Medan'),
('14S22', 'Jemet Store', 'Kalimantan'),
('15S22', 'Serba ada', 'Medan'),
('20S22', 'Your Fashion', 'Jakarta Timur');
```

```

SQLQuery1.sql - ch...CHARLOS\asus (58) ̤ ×

-- Tabel Penjual

CREATE TABLE Penjual (
    ID_Penjual VARCHAR(10) PRIMARY KEY,
    Nama_Toko VARCHAR(50),
    Alamat_Toko VARCHAR(100)
);

select * from Penjual

INSERT INTO Penjual (ID_Penjual, Nama_Toko, Alamat_Toko)
VALUES
('11S22', 'Carlos Shop', 'Jakarta Timur'),
('13S22', 'Mari Belanja', 'Medan'),
('14S22', 'Janet Store', 'Kalimantan'),
('15S22', 'Serba ada', 'Medan'),
('20S22', 'Your Fashion', 'Jakarta Timur');

select * from Penjual

```

99 % ̤ Messages
5 rows affected
Completion time: 2023-11-30T21:00:06.1269699+07:00

99 % ̤ Results ̤ Messages
charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows

Query executed successfully.

Gambar 28. Insert Value into Tabel Penjual

Syntax

```
select * from Penjual
```

```

SQLQuery1.sql - ch...CHARLOS\asus (58) ̤ ×

-- Tabel Penjual

CREATE TABLE Penjual (
    ID_Penjual VARCHAR(10) PRIMARY KEY,
    Nama_Toko VARCHAR(50),
    Alamat_Toko VARCHAR(100)
);

select * from Penjual

INSERT INTO Penjual (ID_Penjual, Nama_Toko, Alamat_Toko)
VALUES
('11S22', 'Carlos Shop', 'Jakarta Timur'),
('13S22', 'Mari Belanja', 'Medan'),
('14S22', 'Janet Store', 'Kalimantan'),
('15S22', 'Serba ada', 'Medan'),
('20S22', 'Your Fashion', 'Jakarta Timur');

select * from Penjual

```

99 % ̤ Results ̤ Messages
charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 5 rows

ID_Penjual	Nama_Toko	Alamat_Toko
1 11S22	Carlos Shop	Jakarta Timur
2 13S22	Mari Belanja	Medan
3 14S22	Janet Store	Kalimantan
4 15S22	Serba ada	Medan
5 20S22	Your Fashion	Jakarta Timur

Query executed successfully.

Gambar 29. Hasil Ouput Insert Value into Tabel Penjual

4.2.1.19 Creating Tabel Pengiriman

Syntax

```
CREATE TABLE Pengiriman (
    No_Resi VARCHAR(10) PRIMARY KEY,
    Jasa_Pengiriman VARCHAR(50),
    Tanggal_Pesanan DATE,
    Status_Pengiriman VARCHAR(50),
    Alamat_Pengiriman VARCHAR(100),
    ID_Penjual VARCHAR(10),
    FOREIGN KEY (ID_Penjual) REFERENCES Penjual(ID_Penjual)
);
```

The screenshot shows the SSMS interface with a query window titled "SQLQuery1.sql - ch...CHARLOS\asus (58)". The code in the window is as follows:

```
-- Tabel Pengiriman
CREATE TABLE Pengiriman (
    No_Resi VARCHAR(10) PRIMARY KEY,
    Jasa_Pengiriman VARCHAR(50),
    Tanggal_Pesanan DATE,
    Status_Pengiriman VARCHAR(50),
    Alamat_Pengiriman VARCHAR(100),
    ID_Penjual VARCHAR(10),
    FOREIGN KEY (ID_Penjual) REFERENCES Penjual(ID_Penjual)
);

select * from Pengiriman

--INSERT INTO Pengiriman (No_Resi, Jasa_Pengiriman, Tanggal_Pesanan, Status_Pengiriman, Alamat_Pengiriman, ID_Penjual)
VALUES
('JNT01', 'JNT', '2023-01-01', 'Dikemas', 'Medan', '11S22'),
('JNT02', 'JNT', '2023-01-01', 'Dalam Perjalanan', 'Medan', '13S22'),
('JNT03', 'JNT', '2023-01-01', 'Dalam Perjalanan', 'Tarutung', '14S22'),
('JNT04', 'JNT', '2023-01-01', 'Dikemas', 'Balige', '15S22'),
('JNT05', 'JNT', '2023-01-01', 'Dikemas', 'Medan', '20S22');
```

The status bar at the bottom indicates "99 %", "Messages", "Commands completed successfully.", "Completion time: 2023-11-30T21:06:44.5920361+07:00", and "charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows". A yellow bar at the bottom also says "Query executed successfully."

Gambar 30. Creating Tabel Pengiriman

4.2.1.20 Viewing Tabel Pengiriman

Syntax

```
select * from Pengiriman
```

```

SQLQuery1.sql - ch...CHARLOS\asus (58)  ✎ ×

-- Tabel Pengiriman
CREATE TABLE Pengiriman (
    No_Resi VARCHAR(10) PRIMARY KEY,
    Jasa_Pengiriman VARCHAR(50),
    Tanggal_Pesanan DATE,
    Status_Pengiriman VARCHAR(50),
    Alamat_Pengiriman VARCHAR(100),
    ID_Penjual VARCHAR(10),
    FOREIGN KEY (ID_Penjual) REFERENCES Penjual(ID_Penjual)
);

select * from Pengiriman

INSERT INTO Pengiriman (No_Resi, Jasa_Pengiriman, Tanggal_Pesanan, Status_Pengiriman, Alamat_Pengiriman, ID_Penjual)
VALUES
    ('JNT01', 'JNT', '2023-01-01', 'Dikemas', 'Medan', '11S22'),
    ('JNT02', 'JNT', '2023-01-01', 'Dalam Perjalanan', 'Medan', '13S22'),
    ('JNT03', 'JNT', '2023-01-01', 'Dalam Perjalanan', 'Tarutung', '14S22'),
    ('JNT04', 'JNT', '2023-01-01', 'Dikemas', 'Balige', '15S22'),
    ('JNT05', 'JNT', '2023-01-01', 'Dikemas', 'Medan', '20S22');

```

The screenshot shows the SSMS interface with a query window containing SQL code to create a table 'Pengiriman' and insert five rows of data. The table has columns: No_Resi (Primary Key), Jasa_Pengiriman, Tanggal_Pesanan, Status_Pengiriman, Alamat_Pengiriman, and ID_Penjual. The inserted data includes various tracking numbers, service types, dates, statuses, locations, and seller IDs.

Gambar 31. Hasil Output Creating Tabel Pengiriman

4.2.1.21 Inserting Tabel Pengiriman

Syntax

```

INSERT INTO Pengiriman (No_Resi, Jasa_Pengiriman, Tanggal_Pesanan, Status_Pengiriman,
Alamat_Pengiriman, ID_Penjual)
VALUES
    ('JNT01', 'JNT', '2023-01-01', 'Dikemas', 'Medan', '11S22'),
    ('JNT02', 'JNT', '2023-01-01', 'Dalam Perjalanan', 'Medan', '13S22'),
    ('JNT03', 'JNT', '2023-01-01', 'Dalam Perjalanan', 'Tarutung', '14S22'),
    ('JNT04', 'JNT', '2023-01-01', 'Dikemas', 'Balige', '15S22'),
    ('JNT05', 'JNT', '2023-01-01', 'Dikemas', 'Medan', '20S22');

```

```

SQLQuery1.sql - ch...CHARLOS(asus (58)) ✖
    Tanggal_Pesanan DATE,
    Status_Pengiriman VARCHAR(50),
    Alamat_Pengiriman VARCHAR(100),
    ID_Penjual VARCHAR(10),
    FOREIGN KEY (ID_Penjual) REFERENCES Penjual(ID_Penjual)
);

select * from Pengiriman

--INSERT INTO Pengiriman (No_Resi, Jasa_Pengiriman, Tanggal_Pesanan, Status_Pengiriman, Alamat_Pengiriman, ID_Penjual)
VALUES
('JNT01', 'JNT', '2023-01-01', 'Dikemas', 'Medan', '11S22'),
('JNT02', 'JNT', '2023-01-01', 'Dalam Perjalanan', 'Medan', '13S22'),
('JNT03', 'JNT', '2023-01-01', 'Dalam Perjalanan', 'Tarutung', '14S22'),
('JNT04', 'JNT', '2023-01-01', 'Dikemas', 'Balige', '15S22'),
('JNT05', 'JNT', '2023-01-01', 'Dikemas', 'Medan', '20S22');

select * from Pengiriman

```

99 % 99 %

Messages

(5 rows affected)

Completion time: 2023-11-30T21:08:03.9211974+07:00

99 % 99 %

Query executed successfully.

charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows

Gambar 32. Insert Value into Tabel Pengiriman

Syntax

```
select * from Pengiriman
```

```

SQLQuery1.sql - ch...CHARLOS(asus (58)) ✖
    Tanggal_Pesanan DATE,
    Status_Pengiriman VARCHAR(50),
    Alamat_Pengiriman VARCHAR(100),
    ID_Penjual VARCHAR(10),
    FOREIGN KEY (ID_Penjual) REFERENCES Penjual(ID_Penjual)
);

select * from Pengiriman

--INSERT INTO Pengiriman (No_Resi, Jasa_Pengiriman, Tanggal_Pesanan, Status_Pengiriman, Alamat_Pengiriman, ID_Penjual)
VALUES
('JNT01', 'JNT', '2023-01-01', 'Dikemas', 'Medan', '11S22'),
('JNT02', 'JNT', '2023-01-01', 'Dalam Perjalanan', 'Medan', '13S22'),
('JNT03', 'JNT', '2023-01-01', 'Dalam Perjalanan', 'Tarutung', '14S22'),
('JNT04', 'JNT', '2023-01-01', 'Dikemas', 'Balige', '15S22'),
('JNT05', 'JNT', '2023-01-01', 'Dikemas', 'Medan', '20S22');

select * from Pengiriman

```

99 % 99 %

Results Messages

No_Resi	Jasa_Pengiriman	Tanggal_Pesanan	Status_Pengiriman	Alamat_Pengiriman	ID_Penjual
1	JNT01	2023-01-01	Dikemas	Medan	11S22
2	JNT02	2023-01-01	Dalam Perjalanan	Medan	13S22
3	JNT03	2023-01-01	Dalam Perjalanan	Tarutung	14S22
4	JNT04	2023-01-01	Dikemas	Balige	15S22
5	JNT05	2023-01-01	Dikemas	Medan	20S22

99 % 99 %

Query executed successfully.

charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 5 rows

Gambar 33. Hasil Output Insert Value into Tabel Pengiriman

4.2.1.22 Creating Tabel Produk

Syntax

```
CREATE TABLE Produk (
    ID_Produk VARCHAR(10) PRIMARY KEY,
    Nama_Produk VARCHAR(50),
    Harga VARCHAR(20),
    Kuantitas INT,
    Kategori VARCHAR(50),
    ID_Penjual VARCHAR(10),
    ID_Keranjang VARCHAR(10),
    FOREIGN KEY (ID_Penjual) REFERENCES Penjual(ID_Penjual)
);
```

The screenshot shows the SSMS interface with a query window titled 'SQLQuery1.sql - ch...CHARLOS\asus (58)'. The code in the window is:

```
-- Tabel Produk
CREATE TABLE Produk (
    ID_Produk VARCHAR(10) PRIMARY KEY,
    Nama_Produk VARCHAR(50),
    Harga VARCHAR(20),
    Kuantitas INT,
    Kategori VARCHAR(50),
    ID_Penjual VARCHAR(10),
    ID_Keranjang VARCHAR(10),
    FOREIGN KEY (ID_Penjual) REFERENCES Penjual(ID_Penjual)
);
select * from Produk
INSERT INTO Produk (ID_Produk, Nama_Produk, Harga, Kuantitas, Kategori, ID_Penjual, ID_Keranjang)
VALUES
('11Y51', 'Baju', 'Rp. 75.000', 2, 'Pakaian', '11S22', 'P001'),
('11Y52', 'Celana Pendek', 'Rp. 70.000', 3, 'Pakaian', '13S22', 'P002'),
('11Y53', 'Sendal Gunung', 'Rp. 180.000', 1, 'Sendal', '14S22', 'P003'),
('11Y54', 'Sepatu', 'Rp. 500.000', 1, 'Sepatu', '15S22', 'P004'),
('11Y55', 'Hoodie', 'Rp. 250.000', 1, 'Pakaian', '20S22', 'P005');
```

The status bar at the bottom indicates '99 %' completion, 'Messages' with 'Commands completed successfully.', 'Completion time: 2023-11-30T21:14:42.5531992+07:00', and 'charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows'.

Gambar 34. Creating Tabel Produk

4.2.1.23 Viewing Tabel Produk

Syntax

```
select * from Produk
```

The screenshot shows a SQL query window titled 'SQLQuery1.sql - ch...CHARLOS\asus (58)'. The code creates a table 'Produk' with columns: ID_Produk (primary key), Nama_Produk, Harga, Kuantitas, Kategori, ID_Penjual, and ID_Keranjang. It includes a foreign key constraint for ID_Penjual referencing the 'Penjual' table. Data is inserted into the 'Produk' table with 5 rows of items like Baju, Celana Pendek, Sendal Gunung, Sepatu, and Hoodie, each with its respective details.

```

-- Tabel Produk
CREATE TABLE Produk (
    ID_Produk VARCHAR(10) PRIMARY KEY,
    Nama_Produk VARCHAR(50),
    Harga VARCHAR(20),
    Kuantitas INT,
    Kategori VARCHAR(50),
    ID_Penjual VARCHAR(10),
    ID_Keranjang VARCHAR(10),
    FOREIGN KEY (ID_Penjual) REFERENCES Penjual(ID_Penjual)
);
select * from Produk
INSERT INTO Produk (ID_Produk, Nama_Produk, Harga, Kuantitas, Kategori, ID_Penjual, ID_Keranjang)
VALUES
('11YS1', 'Baju', 'Rp. 75.000', 2, 'Pakaian', '11S22', 'P001'),
('11YS2', 'Celana Pendek', 'Rp. 70.000', 3, 'Pakaian', '13S22', 'P002'),
('11YS3', 'Sendal Gunung', 'Rp. 180.000', 1, 'Sendal', '14S22', 'P003'),
('11YS4', 'Sepatu', 'Rp. 500.000', 1, 'Sepatu', '15S22', 'P004'),
('11YS5', 'Hoodie', 'Rp. 250.000', 1, 'Pakaian', '20S22', 'P005');

```

Gambar 35. Hasil Output Creating Tabel Produk

4.2.1.24 Inserting Tabel Produk

Syntax

```

INSERT INTO Produk (ID_Produk, Nama_Produk, Harga, Kuantitas, Kategori, ID_Penjual,
ID_Keranjang)
VALUES
('11YS1', 'Baju', 'Rp. 75.000', 2, 'Pakaian', '11S22', 'P001'),
('11YS2', 'Celana Pendek', 'Rp. 70.000', 3, 'Pakaian', '13S22', 'P002'),
('11YS3', 'Sendal Gunung', 'Rp. 180.000', 1, 'Sendal', '14S22', 'P003'),
('11YS4', 'Sepatu', 'Rp. 500.000', 1, 'Sepatu', '15S22', 'P004'),
('11YS5', 'Hoodie', 'Rp. 250.000', 1, 'Pakaian', '20S22', 'P005');

```

```

SQLQuery1.sql - ch...CHARLOS\asus (58)  ↗ ×

    ID_Penjual VARCHAR(10),
    ID_Keranjang VARCHAR(10),
    FOREIGN KEY (ID_Penjual) REFERENCES Penjual(ID_Penjual)
);

select * from Produk

INSERT INTO Produk (ID_Produk, Nama_Produk, Harga, Kuantitas, Kategori, ID_Penjual, ID_Keranjang)
VALUES
('11YS1', 'Baju', 'Rp. 75.000', 2, 'Pakaian', '11S22', 'P001'),
('11YS2', 'Celana Pendek', 'Rp. 70.000', 3, 'Pakaian', '13S22', 'P002'),
('11YS3', 'Sendal Gunung', 'Rp. 180.000', 1, 'Sendal', '14S22', 'P003'),
('11YS4', 'Sepatu', 'Rp. 500.000', 1, 'Sepatu', '15S22', 'P004'),
('11YS5', 'Hoodie', 'Rp. 250.000', 1, 'Pakaian', '20S22', 'P005');

select * from Produk

```

99 % ↴ 4 Messages
(5 rows affected)
Completion time: 2023-11-30T21:17:03.8438014+07:00

99 % ↴ 4 Query executed successfully. charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows

Gambar 36. Insert Value into Tabel Produk

Syntax

```
select * from Produk
```

```

SQLQuery1.sql - ch...CHARLOS\asus (58)  ↗ ×

    ID_Penjual VARCHAR(10),
    ID_Keranjang VARCHAR(10),
    FOREIGN KEY (ID_Penjual) REFERENCES Penjual(ID_Penjual)
);

select * from Produk

INSERT INTO Produk (ID_Produk, Nama_Produk, Harga, Kuantitas, Kategori, ID_Penjual, ID_Keranjang)
VALUES
('11YS1', 'Baju', 'Rp. 75.000', 2, 'Pakaian', '11S22', 'P001'),
('11YS2', 'Celana Pendek', 'Rp. 70.000', 3, 'Pakaian', '13S22', 'P002'),
('11YS3', 'Sendal Gunung', 'Rp. 180.000', 1, 'Sendal', '14S22', 'P003'),
('11YS4', 'Sepatu', 'Rp. 500.000', 1, 'Sepatu', '15S22', 'P004'),
('11YS5', 'Hoodie', 'Rp. 250.000', 1, 'Pakaian', '20S22', 'P005');

select * from Produk

```

99 % ↴ 4 Results | Messages

ID_Produk	Nama_Produk	Harga	Kuantitas	Kategori	ID_Penjual	ID_Keranjang
11YS1	Baju	Rp. 75.000	2	Pakaian	11S22	P001
11YS2	Celana Pendek	Rp. 70.000	3	Pakaian	13S22	P002
11YS3	Sendal Gunung	Rp. 180.000	1	Sendal	14S22	P003
11YS4	Sepatu	Rp. 500.000	1	Sepatu	15S22	P004
11YS5	Hoodie	Rp. 250.000	1	Pakaian	20S22	P005

99 % ↴ 4 Query executed successfully. charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 5 rows

Gambar 37. Hasil Output Insert Value into Tabel Produk

4.2.1.25 Creating Tabel Keranjang

Syntax

```
CREATE TABLE Keranjang (
    ID_Keranjang VARCHAR(10) PRIMARY KEY,
    ID_Pelanggan VARCHAR(10),
    Kuantitas INT,
    Catatan VARCHAR(50)
);
```

The screenshot shows the SSMS interface with a query window containing the following SQL code:

```
-- Tabel Keranjang
CREATE TABLE Keranjang (
    ID_Keranjang VARCHAR(10) PRIMARY KEY,
    ID_Pelanggan VARCHAR(10),
    Kuantitas INT,
    Catatan VARCHAR(50)
);

select * from Keranjang

INSERT INTO Keranjang (ID_Keranjang, ID_Pelanggan, Kuantitas, Catatan)
VALUES
('P001', '13522', 2, 'Stok tersedia'),
('P002', '14522', 3, 'Stok habis'),
('P003', '16522', 1, 'Stok tersedia'),
('P004', '11522', 1, 'Stok habis'),
('P005', '12522', 1, 'Stok tersedia');

select * from Keranjang
```

The status bar at the bottom indicates "Query executed successfully." and "0 rows".

Gambar 38. Creating Tabel Keranjang

4.2.1.26 Viewing Tabel Keranjang

Syntax

```
select * from Keranjang
```

The screenshot shows the SQL Query Editor window in SQL Server Management Studio. The query being run is:

```
-- Tabel Keranjang
CREATE TABLE Keranjang (
    ID_Keranjang VARCHAR(10) PRIMARY KEY,
    ID_Pelanggan VARCHAR(10),
    Kuantitas INT,
    Catatan VARCHAR(50)
);

select * from Keranjang

INSERT INTO Keranjang (ID_Keranjang, ID_Pelanggan, Kuantitas, Catatan)
VALUES
('P001', '13S22', 2, 'Stok tersedia'),
('P002', '14S22', 3, 'Stok habis'),
('P003', '16S22', 1, 'Stok tersedia'),
('P004', '11S22', 1, 'Stok habis'),
('P005', '12S22', 1, 'Stok tersedia');

select * from Keranjang
```

The Results tab shows the following table structure:

ID_Keranjang	ID_Pelanggan	Kuantitas	Catatan

At the bottom of the results pane, it says "Query executed successfully." and shows the status bar with "charlos (15.0 RTM) CHARLOS\asus (58) Proyek_Basdat_Kelompok01 00:00:00 0 rows".

Gambar 39. Hasil Output Creating Tabel Keranjang

4.2.1.27 Inserting Tabel Keranjang

Syntax

```
INSERT INTO Keranjang (ID_Keranjang, ID_Pelanggan, Kuantitas, Catatan)
VALUES
('P001', '13S22', 2, 'Stok tersedia'),
('P002', '14S22', 3, 'Stok habis'),
('P003', '16S22', 1, 'Stok tersedia'),
('P004', '11S22', 1, 'Stok habis'),
('P005', '12S22', 1, 'Stok tersedia');
```

```

SQLQuery1.sql - ch...CHARLOS(asus (58))  ↗ X
-- Tabel Keranjang
CREATE TABLE Keranjang (
    ID_Keranjang VARCHAR(10) PRIMARY KEY,
    ID_Pelanggan VARCHAR(10),
    Kuantitas INT,
    Catatan VARCHAR(50)
);
select * from Keranjang

INSERT INTO Keranjang (ID_Keranjang, ID_Pelanggan, Kuantitas, Catatan)
VALUES
    ('P001', '13S22', 2, 'Stok tersedia'),
    ('P002', '14S22', 3, 'Stok habis'),
    ('P003', '16S22', 1, 'Stok tersedia'),
    ('P004', '11S22', 1, 'Stok habis'),
    ('P005', '12S22', 1, 'Stok tersedia');

select * from Keranjang

99 %  ↵
Messages
(5 rows affected)
Completion time: 2023-11-30T21:25:33.1843680+07:00

99 %  ↵
Query executed successfully. charlos (15.0 RTM) CHARLOS(asus (58)) Proyek_Basdat_Kelompok01 00:00:00 0 rows

```

Gambar 40. Insert Value into Tabel Keranjang

Syntax

```
select * from Keranjang
```

```

SQLQuery1.sql - ch...CHARLOS(asus (58))  ↗ X
-- Tabel Keranjang
CREATE TABLE Keranjang (
    ID_Keranjang VARCHAR(10) PRIMARY KEY,
    ID_Pelanggan VARCHAR(10),
    Kuantitas INT,
    Catatan VARCHAR(50)
);
select * from Keranjang

INSERT INTO Keranjang (ID_Keranjang, ID_Pelanggan, Kuantitas, Catatan)
VALUES
    ('P001', '13S22', 2, 'Stok tersedia'),
    ('P002', '14S22', 3, 'Stok habis'),
    ('P003', '16S22', 1, 'Stok tersedia'),
    ('P004', '11S22', 1, 'Stok habis'),
    ('P005', '12S22', 1, 'Stok tersedia');

select * from Keranjang

99 %  ↵
Results  ↵ Messages
ID_Keranjang ID_Pelanggan Kuantitas Catatan
1 P001         13S22        2 Stok tersedia
2 P002         14S22        3 Stok habis
3 P003         16S22        1 Stok tersedia
4 P004         11S22        1 Stok habis
5 P005         12S22        1 Stok tersedia

99 %  ↵
Query executed successfully. charlos (15.0 RTM) CHARLOS(asus (58)) Proyek_Basdat_Kelompok01 00:00:00 5 rows

```

Gambar 41. Hasil Output Insert Value into Tabel Keranjang

4.2.1.28 Creating Tabel Tinjauan

Syntax

```
CREATE TABLE Tinjauan (
    ID_Tinjauan VARCHAR(10) PRIMARY KEY,
    Isi_Tinjauan VARCHAR(100),
    Tanggal_Tinjauan DATE,
    Rating INT,
    ID_Produk VARCHAR(10),
    ID_Pelanggan VARCHAR(10),
    FOREIGN KEY (ID_Produk) REFERENCES Produk(ID_Produk),
    FOREIGN KEY (ID_Pelanggan) REFERENCES Pelanggan(ID_Pelanggan)
);
```

The screenshot shows the SQL Query Editor in SSMS with the following code:

```
-- Tabel Tinjauan
CREATE TABLE Tinjauan (
    ID_Tinjauan VARCHAR(10) PRIMARY KEY,
    Isi_Tinjauan VARCHAR(100),
    Tanggal_Tinjauan DATE,
    Rating INT,
    ID_Produk VARCHAR(10),
    ID_Pelanggan VARCHAR(10),
    FOREIGN KEY (ID_Produk) REFERENCES Produk(ID_Produk),
    FOREIGN KEY (ID_Pelanggan) REFERENCES Pelanggan(ID_Pelanggan)
);

select * from Tinjauan

INSERT INTO Tinjauan (ID_Tinjauan, Isi_Tinjauan, Tanggal_Tinjauan, Rating, ID_Produk, ID_Pelanggan)
VALUES
('001T1', 'Kualitas barang baik', '2023-01-06', 4, '11YS1', '13S22'),
('001T2', 'Kualitas barang sangat baik', '2023-01-07', 5, '11YS2', '14S22'),
('001T3', 'Kualitas barang baik', '2023-01-08', 3, '11YS3', '16S22'),
('001T4', 'Kualitas barang kurang baik', '2023-01-09', 2, '11YS4', '11S22'),
```

The Messages pane shows "Commands completed successfully." and the completion time: 2023-11-30T21:39:11.6614025+07:00.

The status bar at the bottom right indicates: charlos (15.0 RTM) | CHARLOS\asus (58) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows.

Gambar 42. Creating Tabel Tinjauan

4.2.1.29 Viewing Tabel Tinjauan

Syntax

```
select * from Tinjauan
```

The screenshot shows the SQL Query window in SSMS. The code is as follows:

```
-- Tabel Tinjauan
CREATE TABLE Tinjauan (
    ID_Tinjauan VARCHAR(10) PRIMARY KEY,
    Isi_Tinjauan VARCHAR(100),
    Tanggal_Tinjauan DATE,
    Rating INT,
    ID_Produk VARCHAR(10),
    ID_Pelanggan VARCHAR(10),
    FOREIGN KEY (ID_Produk) REFERENCES Produk(ID_Produk),
    FOREIGN KEY (ID_Pelanggan) REFERENCES Pelanggan(ID_Pelanggan)
);

select * from Tinjauan

-- INSERT INTO Tinjauan (ID_Tinjauan, Isi_Tinjauan, Tanggal_Tinjauan, Rating, ID_Produk, ID_Pelanggan)
VALUES
('001T1', 'Kualitas barang baik', '2023-01-06', 4, '11YS1', '13S22'),
('001T2', 'Kualitas barang sangat baik', '2023-01-07', 5, '11YS2', '14S22'),
('001T3', 'Kualitas barang baik', '2023-01-08', 3, '11YS3', '16S22'),
('001T4', 'Kualitas barang kurang baik', '2023-01-09', 2, '11YS4', '11S22'),
('001T5', 'Kualitas barang baik', '2023-01-10', 4, '11YS5', '12S22'),
```

The results pane is empty. At the bottom, a message bar says "Query executed successfully.".

Gambar 43. Hasil Output Creating Tabel Tinjauan

4.2.1.30 Inserting Tabel Tinjauan

Syntax

```
INSERT INTO Tinjauan (ID_Tinjauan, Isi_Tinjauan, Tanggal_Tinjauan, Rating, ID_Produk,
ID_Pelanggan)
VALUES
('001T1', 'Kualitas barang baik', '2023-01-06', 4, '11YS1', '13S22'),
('001T2', 'Kualitas barang sangat baik', '2023-01-07', 5, '11YS2', '14S22'),
('001T3', 'Kualitas barang baik', '2023-01-08', 3, '11YS3', '16S22'),
('001T4', 'Kualitas barang kurang baik', '2023-01-09', 2, '11YS4', '11S22'),
('001T5', 'Kualitas barang baik', '2023-01-10', 4, '11YS5', '12S22');
```

```

SQLQuery1.sql - ch...CHARLOS(asus (58))* ✎ ×
Isi_Tinjauan VARCHAR(100),
Tanggal_Tinjauan DATE,
Rating INT,
ID_Produk VARCHAR(10),
ID_Pelanggan VARCHAR(10),
FOREIGN KEY (ID_Produk) REFERENCES Produk(ID_Produk),
FOREIGN KEY (ID_Pelanggan) REFERENCES Pelanggan(ID_Pelanggan)
);

select * from Tinjauan

INSERT INTO Tinjauan (ID_Tinjauan, Isi_Tinjauan, Tanggal_Tinjauan, Rating, ID_Produk, ID_Pelanggan)
VALUES
('001T1', 'Kualitas barang baik', '2023-01-06', 4, '11YS1', '13S22'),
('001T2', 'Kualitas barang sangat baik', '2023-01-07', 5, '11YS2', '14S22'),
('001T3', 'Kualitas barang baik', '2023-01-08', 3, '11YS3', '16S22'),
('001T4', 'Kualitas barang kurang baik', '2023-01-09', 2, '11YS4', '11S22'),
('001T5', 'Kualitas barang baik', '2023-01-10', 4, '11YS5', '12S22');

select * from Tinjauan

```

Messages

(5 rows affected)

Completion time: 2023-12-01T01:02:54.9087433+07:00

99 %

Query executed successfully.

Gambar 44. Insert Value intoTabel Tinjauan

Syntax

```
select * from Tinjauan
```

```

SQLQuery1.sql - ch...CHARLOS(asus (58))* ✎ ×
Isi_Tinjauan VARCHAR(100),
Tanggal_Tinjauan DATE,
Rating INT,
ID_Produk VARCHAR(10),
ID_Pelanggan VARCHAR(10),
FOREIGN KEY (ID_Produk) REFERENCES Produk(ID_Produk),
FOREIGN KEY (ID_Pelanggan) REFERENCES Pelanggan(ID_Pelanggan)
);

select * from Tinjauan

INSERT INTO Tinjauan (ID_Tinjauan, Isi_Tinjauan, Tanggal_Tinjauan, Rating, ID_Produk, ID_Pelanggan)
VALUES
('001T1', 'Kualitas barang baik', '2023-01-06', 4, '11YS1', '13S22'),
('001T2', 'Kualitas barang sangat baik', '2023-01-07', 5, '11YS2', '14S22'),
('001T3', 'Kualitas barang baik', '2023-01-08', 3, '11YS3', '16S22'),
('001T4', 'Kualitas barang kurang baik', '2023-01-09', 2, '11YS4', '11S22'),
('001T5', 'Kualitas barang baik', '2023-01-10', 4, '11YS5', '12S22');

select * from Tinjauan

```

Results

ID_Tinjauan	Isi_Tinjauan	Tanggal_Tinjauan	Rating	ID_Produk	ID_Pelanggan	
1	001T1	Kualitas barang baik	2023-01-06	4	11YS1	13S22
2	001T2	Kualitas barang sangat baik	2023-01-07	5	11YS2	14S22
3	001T3	Kualitas barang baik	2023-01-08	3	11YS3	16S22
4	001T4	Kualitas barang kurang baik	2023-01-09	2	11YS4	11S22
5	001T5	Kualitas barang baik	2023-01-10	4	11YS5	12S22

99 %

Messages

Query executed successfully.

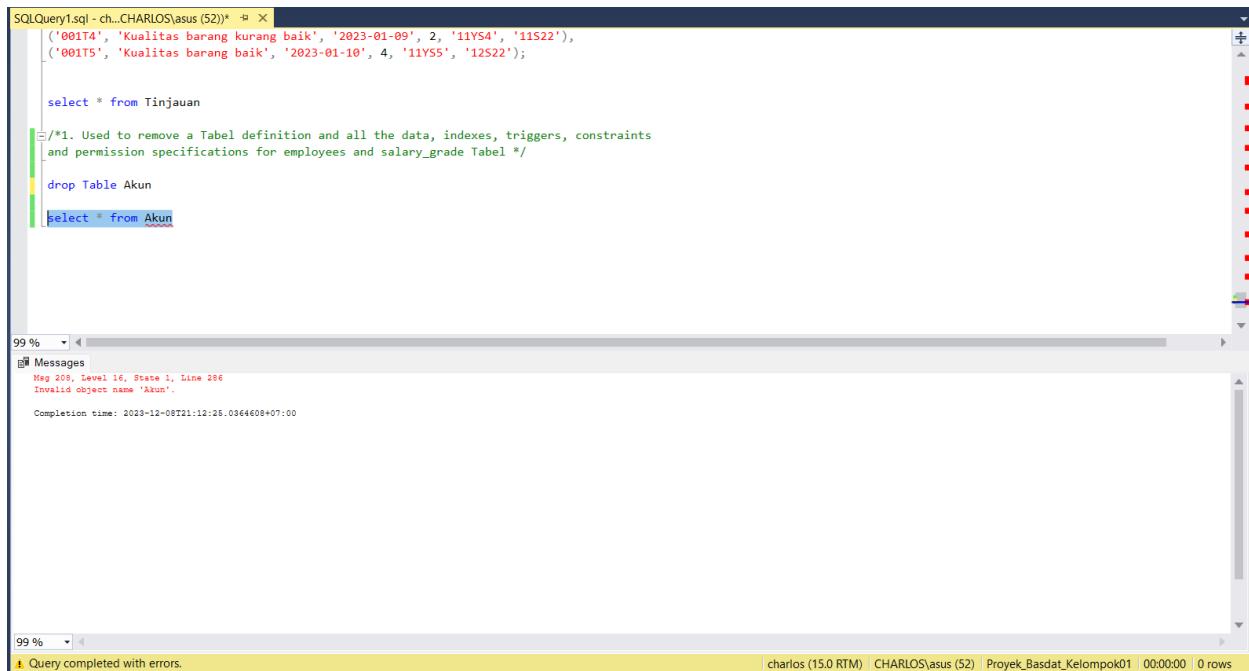
Gambar 45. Hasil Output Insert Value intoTabel Tinjauan

4.3 Creating Basic Query (Select, Update, Set Operators, Aggregate Function, Null Value)

Syntax

```
/*1. Used to remove a Tabel definition and all the data, indexes, triggers, constraints  
and permission specifications for employees and salary_grade Tabel */  
  
drop Table Akun  
  
select * from Akun
```

Output:



The screenshot shows a SQL Server Management Studio (SSMS) window. In the main pane, there is a SQL script:

```
SQLQuery1.sql - ch...CHARLOS\asus (52)*  # X  
/*1. Used to remove a Tabel definition and all the data, indexes, triggers, constraints  
and permission specifications for employees and salary_grade Tabel */  
  
drop Table Akun  
  
select * from Akun
```

In the bottom-left corner of the main pane, there is a small error message:

Msg 208, Level 16, State 1, Line 286
Invalid object name 'Akun'.

Below the main pane, the status bar displays:

Completion time: 2023-12-08T21:12:25.0364608+07:00

At the very bottom of the screen, a yellow bar indicates:

Query completed with errors.

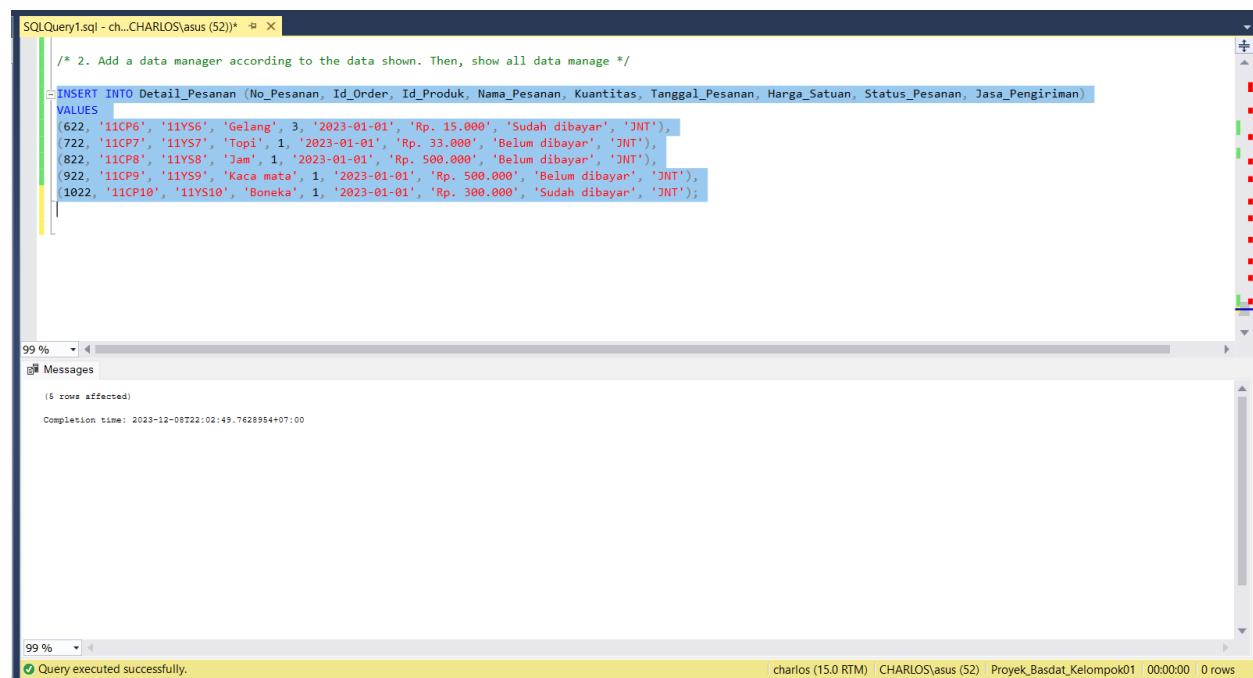
Gambar 46. Melakukan Drop Tabel Akun

Syntax

```
/* 2. Add a data manager according to the data shown. Then, show all data manage */

INSERT INTO Detail_Pesanan (No_Pesanan, Id_Order, Id_Produk, Nama_Pesanan, Kuantitas,
Tanggal_Pesanan, Harga_Satuan, Status_Pesanan, Jasa_Pengiriman)
VALUES
(622, '11CP6', '11YS6', 'Gelang', 3, '2023-01-01', 'Rp. 15.000', 'Sudah dibayar', 'JNT'),
(722, '11CP7', '11YS7', 'Topi', 1, '2023-01-01', 'Rp. 33.000', 'Belum dibayar', 'JNT'),
(822, '11CP8', '11YS8', 'Jam', 1, '2023-01-01', 'Rp. 500.000', 'Belum dibayar', 'JNT'),
(922, '11CP9', '11YS9', 'Kaca mata', 1, '2023-01-01', 'Rp. 500.000', 'Belum dibayar',
'JNT'),
(1022, '11CP10', '11YS10', 'Boneka', 1, '2023-01-01', 'Rp. 300.000', 'Sudah dibayar',
'JNT');
```

Execute



The screenshot shows a SQL query window in SSMS. The query is identical to the one shown in the previous code block. It inserts 10 rows into the 'Detail_Pesanan' table. The execution completed successfully with 6 rows affected.

```
SQLQuery1.sql - ch...CHARLOS\asus (52)*  X
/*
2. Add a data manager according to the data shown. Then, show all data manage */

INSERT INTO Detail_Pesanan (No_Pesanan, Id_Order, Id_Produk, Nama_Pesanan, Kuantitas,
Tanggal_Pesanan, Harga_Satuan, Status_Pesanan, Jasa_Pengiriman)
VALUES
(622, '11CP6', '11YS6', 'Gelang', 3, '2023-01-01', 'Rp. 15.000', 'Sudah dibayar', 'JNT'),
(722, '11CP7', '11YS7', 'Topi', 1, '2023-01-01', 'Rp. 33.000', 'Belum dibayar', 'JNT'),
(822, '11CP8', '11YS8', 'Jam', 1, '2023-01-01', 'Rp. 500.000', 'Belum dibayar', 'JNT'),
(922, '11CP9', '11YS9', 'Kaca mata', 1, '2023-01-01', 'Rp. 500.000', 'Belum dibayar',
'JNT'),
(1022, '11CP10', '11YS10', 'Boneka', 1, '2023-01-01', 'Rp. 300.000', 'Sudah dibayar',
'JNT');
```

99 % 99 %

Messages

(6 rows affected)

Completion time: 2023-12-08T22:02:49.7628954+07:00

99 % 99 %

Query executed successfully.

charlos (15.0 RTM) | CHARLOS\asus (52) | Proyek_Basdat_Kelompok01 | 00:00:00 | 0 rows

Gambar 47. Insert Value Data ke dalam Tabel Detail Pesanan

Output

The screenshot shows the SQL Query Editor window with the following content:

```
SQLQuery1.sql - ch...CHARLOS(asus (52)) * 
/*
 2. Add a data manager according to the data shown. Then, show all data manage */
INSERT INTO Detail_Pesanan (No_Pesanan, Id_Order, Id_Produk, Nama_Pesanan, Kuantitas, Tanggal_Pesanan, Harga_Satuan, Status_Pesanan, Jasa_Pengiriman)
VALUES
(622, '11CP6', '11YS6', 'Gelang', 3, '2023-01-01', 'Rp. 15.000', 'Sudah dibayar', 'JNT'),
(722, '11CP7', '11YS7', 'Topi', 1, '2023-01-01', 'Rp. 33.000', 'Belum dibayar', 'JNT'),
(822, '11CP8', '11YS8', 'Jam', 1, '2023-01-01', 'Rp. 500.000', 'Belum dibayar', 'JNT'),
(922, '11CP9', '11YS9', 'Kaca mata', 1, '2023-01-01', 'Rp. 500.000', 'Belum dibayar', 'JNT'),
(1022, '11CP10', '11YS10', 'Boneka', 1, '2023-01-01', 'Rp. 300.000', 'Sudah dibayar', 'JNT');
```

The Results tab displays the inserted data in a table:

No_Pesanan	Id_Order	Id_Produk	Nama_Pesanan	Kuantitas	Tanggal_Pesanan	Harga_Satuan	Status_Pesanan	Jasa_Pengiriman	
1	122	11CP1	11YS1	Baju	2	2023-01-01	Rp. 75.000	Sudah dibayar	JNT
2	222	11CP2	11YS2	Celana Pendek	3	2023-01-01	Rp. 70.000	Belum dibayar	JNT
3	322	11CP3	11YS3	Sandal Gunung	1	2023-01-01	Rp. 180.000	Belum dibayar	JNT
4	422	11CP4	11YS4	Sepatu	1	2023-01-01	Rp. 500.000	Belum dibayar	JNT
5	522	11CP5	11YS5	Hoodie	1	2023-01-01	Rp. 250.000	Sudah dibayar	JNT
6	622	11CP6	11YS6	Gelang	3	2023-01-01	Rp. 15.000	Sudah dibayar	JNT
7	722	11CP7	11YS7	Topi	1	2023-01-01	Rp. 33.000	Belum dibayar	JNT
8	822	11CP8	11YS8	Jam	1	2023-01-01	Rp. 500.000	Belum dibayar	JNT
9	922	11CP9	11YS9	Kaca mata	1	2023-01-01	Rp. 500.000	Belum dibayar	JNT
10	1022	11CP10	11YS10	Boneka	1	2023-01-01	Rp. 300.000	Sudah dibayar	JNT

At the bottom, a message indicates the query was executed successfully.

Gambar 48. Hasil Output Data yang Diinput ke dalam Tabel Detail Pesanan

Syntax

```
/* 3. Change the Pelanggan data which has Id_Pelanggan is 13S22! Show all
of user identity in the Pelanggan Tabel! */

UPDATE Pelanggan
SET nama = 'Andika'
WHERE Id_Pelanggan = '13S22'

SELECT * FROM Pelanggan
```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS\asus (52)". The query is:

```

/* 3. Change the Pelanggan data which has Id_Pelanggan is 13S22! Show all
of user identity in the Pelanggan Tabel! */

UPDATE Pelanggan
SET nama = 'Andika'
WHERE Id_Pelanggan = '13522'

SELECT * FROM Pelanggan

```

The results pane displays a table with columns "Id_Pelanggan", "Nama", and "Alamat". The data is as follows:

	Id_Pelanggan	Nama	Alamat
1	11S22	Lena	Medan
2	12S22	Dila	Sibolga
3	13S22	Andika	Medan
4	14S22	Yessi	Tarutung
5	16S22	Anton	Balige
6	21S22	David	Medan
7	22S22	Andre	Tarutung
8	23S22	Tian	Balige
9	24S22	Johan	Medan
10	25S22	Mitha	Sibolga

A message at the bottom of the results pane says "Query executed successfully." and indicates "charios (15.0 RTM) | CHARLOS\asus (52) | Proyek_Basdat_Kelompok01 | 00:00:00 | 10 rows".

Gambar 49. Hasil Execute Mengupdate Salah Satu Nama Pelanggan

Syntax

```

/* 4. Displays data on Pesanan items whose price is below 200,000 and has a name ending
with the letter "G" character!*/

SELECT Nama_Pesanan, Harga_Satuan
FROM Detail_Pesanan
WHERE Nama_Pesanan LIKE '%G' AND Harga_Satuan < 'Rp. 200.000';

```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS(asus (52))". The query is:

```
SELECT * FROM Pelanggan
/* 4. Displays data on ordered items whose price is below 200,000 and has a name ending
with the letter "G" character*/
SELECT Nama_Pesanan, Harga_Satuan
FROM Detail_Pesanan
WHERE Nama_Pesanan LIKE '%G' AND Harga_Satuan < 'Rp. 200.000';
```

The results pane shows the following data:

	Nama_Pesanan	Harga_Satuan
1	Sandal Gunung	Rp. 180.000
2	Gelang	Rp. 15.000

At the bottom, a message says "Query executed successfully." and shows session details: charlos (15.0 RTM) | CHARLOS\asus (52) | Proyek_Basdat_Kelompok01 | 00:00:00 | 2 rows.

Gambar 50. Displays Pesanan Item Data That Has an Nama Pesanan with the Final Letter 'G' and Harga Pesanan > 200,000

Syntax

```
/* 5. Change the Detail Pesanan data which has No_pesanan is 622! Show all
of Pesanan in the Detail Pesanan Tabel! */

UPDATE Detail_Pesanan
SET No_Pesanan = '622'
WHERE No_Pesanan = '622'
SELECT * FROM Detail_Pesanan
```

The screenshot shows a SQL query window with the following code:

```

SELECT Nama_Pesanan, Harga_Satuan
FROM Detail_Pesanan
WHERE Nama_Pesanan LIKE '%G' AND Harga_Satuan < 'Rp. 200.000';

/* 5. Change the Detail Pesanan data which has No_pesanan is 622! Show all
of Pesanan in the Detail Pesanan Tabel */

UPDATE Detail_Pesanan
SET No_Pesanan = '622'
WHERE No_Pesanan = '622';
SELECT * FROM Detail_Pesanan

```

The results pane displays a table with 10 rows of data from the 'Detail_Pesanan' table. The columns are: No_Pesanan, Id_Order, Id_Produk, Nama_Pesanan, Kuantitas, Tanggal_Pesanan, Harga_Satuan, Status_Pesanan, and Jasa_Pengiriman.

No_Pesanan	Id_Order	Id_Produk	Nama_Pesanan	Kuantitas	Tanggal_Pesanan	Harga_Satuan	Status_Pesanan	Jasa_Pengiriman
122	11CP1	11YS1	Baju	2	2023-01-01	Rp. 75.000	Sudah dibayar	JNT
222	11CP2	11YS2	Celana Pendek	3	2023-01-01	Rp. 70.000	Belum dibayar	JNT
322	11CP3	11YS3	Sandal Gunung	1	2023-01-01	Rp. 180.000	Belum dibayar	JNT
422	11CP4	11YS4	Sepatu	1	2023-01-01	Rp. 500.000	Belum dibayar	JNT
522	11CP5	11YS5	Hoodie	1	2023-01-01	Rp. 250.000	Sudah dibayar	JNT
622	11CP6	11YS6	Gelang	3	2023-01-01	Rp. 15.000	Sudah dibayar	JNT
722	11CP7	11YS7	Topi	1	2023-01-01	Rp. 33.000	Belum dibayar	JNT
822	11CP8	11YS8	Jam	1	2023-01-01	Rp. 500.000	Belum dibayar	JNT
922	11CP9	11YS9	Kaca mata	1	2023-01-01	Rp. 500.000	Belum dibayar	JNT
1022	11CP10	11YS10	Boneka	1	2023-01-01	Rp. 300.000	Sudah dibayar	JNT

At the bottom of the results pane, it says "Query executed successfully." and shows the session details: charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 10 rows.

Gambar 51. Update Data From Table Detail Pesanan

Syntax

```

/*6. Add status column to the Pelanggan Tabel which has data type varchar */
ALTER Table Pelanggan
ADD Aktivitas VARCHAR (50);

SELECT * FROM Pelanggan

```

The screenshot shows a SQL query window with the following code:

```

SQLQuery1.sql - ch...CHARLOS\asus (69)*  □ X
of Pesanan in the Detail Pesanan Tabel! */

--UPDATE Detail_Pesanan
SET No_Pesanan = '622'
WHERE No_Pesanan = '622'
SELECT * FROM Detail_Pesanan

/*6. Add status column to the Pelanggan Tabel which has data type varchar */
ALTER Table Pelanggan
ADD Aktivitas VARCHAR (50);

SELECT * FROM Pelanggan

```

The results pane displays the contents of the 'Pelanggan' table:

	Id_Pelanggan	Nama	Alamat	Aktivitas
1	11S22	Lena	Medan	NULL
2	12S22	Dilan	Sibolga	NULL
3	13S22	Andika	Medan	NULL
4	14S22	Yessi	Tarutung	NULL
5	16S22	Anton	Balige	NULL
6	21S22	David	Medan	NULL
7	22S22	Andre	Tarutung	NULL
8	23S22	Tian	Balige	NULL
9	24S22	Johan	Medan	NULL
10	25S22	Mitha	Sibolga	NULL

At the bottom, a message indicates the query was executed successfully.

Gambar 52. Add Aktivitas Column ke dalam Table Pelanggan

Syntax

```

/*7. Deleting the Aktivitas column in the Pelanggan Tabel */
ALTER Table Pelanggan
DROP COLUMN Aktivitas;

SELECT * FROM Pelanggan

```

```

SQLQuery1.sql - ch..CHARLOS\asus (69)  □ X
/*
6. Add status column to the Pelanggan Tabel which has data type varchar */
ALTER Table Pelanggan
ADD Aktivitas VARCHAR (50);

SELECT * FROM Pelanggan

/*
7. Deleting the Aktivitas column in the Pelanggan Tabel */
ALTER Table Pelanggan
DROP COLUMN Aktivitas;

SELECT * FROM Pelanggan

```

Results

	Id_Pelanggan	Nama	Alamat
1	11922	Lena	Medan
2	12522	Dilan	Sibolga
3	13522	Andika	Medan
4	14522	Yessi	Tarutung
5	16522	Anton	Balige
6	21522	David	Medan
7	22522	Andre	Tarutung
8	23522	Tian	Balige
9	24522	Johan	Medan
10	25522	Miha	Sibolga

Query executed successfully.

Gambar 53. Deleting Aktivitas Column Pada Table Pelanggan

Syntax

```

/*8. Used to perform operations such as addition to Pesanan Prices */

SELECT
    No_Pesanan,
    Id_Order,
    Id_Produk,
    Nama_Pesanan,
    Kuantitas,
    Tanggal_Pesanan,
    Harga_Satuan AS 'Harga Satuan',
    CONCAT('Rp. ', FORMAT(CAST(REPLACE(REPLACE(Harga_Satuan, 'Rp. ', ''), '.', '')) +
10000 AS INT), 'N0')) AS 'Jumlah Harga'
FROM
    Detail_Pesanan;

```

```

SQLQuery1.sql - ch..CHARLOS\asus (69) 99 %
/*8. Used to perform operations such as addition to Pesanan Prices */

SELECT
    No_Pesanan,
    Id_Order,
    Id_Prodук,
    Nama_Pesanan,
    Kuantitas,
    Tanggal_Pesanan,
    Harga_Satuan AS 'Harga Satuan',
    CONCAT('Rp. ', FORMAT(CAST(REPLACE(REPLACE(Harga_Satuan, 'Rp. ', ''), ',', '') + 10000 AS INT), '#0')) AS 'Jumlah Harga'
FROM
    Detail_Pesanan;

```

No_Pesanan	Id_Order	Id_Prodук	Nama_Pesanan	Kuantitas	Tanggal_Pesanan	Harga_Satuan	Jumlah Harga
122	11CP1	11YS1	Baju	2	2023-01-01	Rp. 75.000	Rp. 85.000
222	11CP2	11YS2	Celana Pendek	3	2023-01-01	Rp. 70.000	Rp. 80.000
322	11CP3	11YS3	Sandal Gunung	1	2023-01-01	Rp. 180.000	Rp. 190.000
422	11CP4	11YS4	Sepatu	1	2023-01-01	Rp. 500.000	Rp. 510.000
522	11CP5	11YS5	Hoodie	1	2023-01-01	Rp. 250.000	Rp. 260.000
622	11CP6	11YS6	Gelang	3	2023-01-01	Rp. 15.000	Rp. 25.000
722	11CP7	11YS7	Topi	1	2023-01-01	Rp. 33.000	Rp. 43.000
822	11CP8	11YS8	Jam	1	2023-01-01	Rp. 500.000	Rp. 510.000
922	11CP9	11YS9	Kaca mata	1	2023-01-01	Rp. 500.000	Rp. 510.000
1022	11CP10	11YS10	Boneka	1	2023-01-01	Rp. 300.000	Rp. 310.000

Query executed successfully.

Gambar 54. Penambahan Harga Sebesar Rp. 10.000 Pada Kolom Harga Satuan

Syntax

```

/*9. Multiplication between the Kuantitas column and the Harga Satuan to get the total
harga*/

SELECT
    No_Pesanan,
    Id_Order,
    Id_Prodук,
    Nama_Pesanan,
    Kuantitas AS 'Kuantitas',
    Tanggal_Pesanan,
    Harga_Satuan AS 'Harga_Satuan',
    CONCAT('Rp. ', FORMAT(Kuantitas * CAST(REPLACE(REPLACE(Harga_Satuan, 'Rp. ', ''),
    '.', '') AS INT), '#0')) AS 'Total Harga'
FROM
    Detail_Pesanan;

```

```

SQLQuery1.sql - ch...CHARLOS\asus (69)* 99 %
/*9. Multiplication between the Kuantitas column and the Harga Satuan to get the total harga*/
SELECT
    No_Pesanan,
    Id_Order,
    Id_Prod,
    Nama_Pesanan,
    Kuantitas AS 'Kuantitas',
    Tanggal_Pesanan,
    Harga_Satuan AS 'Harga_Satuan',
    CONCAT('Rp. ', FORMAT(Kuantitas * CAST(REPLACE(REPLACE(Harga_Satuan, 'Rp. ', ''), ',', '.') AS INT), 'N0')) AS 'Total_Harga'
FROM
    Detail_Pesanan;

```

No_Pesanan	Id_Order	Id_Prod	Nama_Pesanan	Kuantitas	Tanggal_Pesanan	Harga_Satuan	Total_Harga
122	11CP1	11YS1	Baju	2	2023-01-01	Rp. 75.000	Rp. 150.000
222	11CP2	11YS2	Celana Pendek	3	2023-01-01	Rp. 70.000	Rp. 210.000
322	11CP3	11YS3	Sandal Gunung	1	2023-01-01	Rp. 180.000	Rp. 180.000
442	11CP4	11YS4	Sepatu	1	2023-01-01	Rp. 500.000	Rp. 500.000
552	11CP5	11YS5	Hoodie	1	2023-01-01	Rp. 250.000	Rp. 250.000
662	11CP6	11YS6	Gelang	3	2023-01-01	Rp. 15.000	Rp. 45.000
772	11CP7	11YS7	Topi	1	2023-01-01	Rp. 33.000	Rp. 33.000
882	11CP8	11YS8	Jam	1	2023-01-01	Rp. 500.000	Rp. 500.000
992	11CP9	11YS9	Kaca mata	1	2023-01-01	Rp. 500.000	Rp. 500.000
101022	11CP10	11YS10	Boneka	1	2023-01-01	Rp. 300.000	Rp. 300.000

Query executed successfully. charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 10 rows

Gambar 55. Perkalian antara Kolom kuantitas dengan Harga satuan untuk memperoleh Total Harga

Syntax

```

/*10. Show No_Pesanan, Id_Prod, Nama_Pesanan, and Kuantitas that has a Kuantitas above 2 */
SELECT
    No_Pesanan,
    Id_Prod,
    Nama_Pesanan,
    Kuantitas
FROM
    Detail_Pesanan
WHERE
    Kuantitas > 2;

```

```

SQLQuery1.sql - ch...CHARLOS\asus (69)  X

/*10. Show No_Pesanan, Id_Produk, Nama_Pesanan, and Kuantitas that has a Kuantitas above 2 */
SELECT
    No_Pesanan,
    Id_Produk,
    Nama_Pesanan,
    Kuantitas
FROM
    Detail_Pesanan
WHERE
    Kuantitas > 2;

```

No_Pesanan	Id_Produk	Nama_Pesanan	Kuantitas
1 222	11YS2	Celana Pendek	3
2 622	11YS6	Gelang	3

99 % ▶

Results Messages

Query executed successfully. charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 2 rows

Gambar 56. Menampilkan data Pesanan dari Detail Pesanan yang memiliki kuantitas di atas 3

Syntax

```

/*11. Tampilkan Id_Produk dan Nama_Pesanan dari tabel Produk. Urutkan data berdasarkan id paling awal! */
SELECT
    P.ID_Produk,
    DP.Nama_Pesanan
FROM
    Produk P
LEFT JOIN
    Detail_Pesanan DP ON P.ID_Produk = DP.Id_Produk
ORDER BY
    P.ID_Produk ASC;

```

The screenshot shows a SQL query window in SQL Server Management Studio. The query retrieves product IDs and names from the 'Produk' table, joined with the 'Detail_Pesanan' table, where the quantity is greater than 2, ordered by product ID. The results are displayed in a table with columns 'ID_Produk' and 'Nama_Pesanan'. The output shows five rows of data.

```
SQLQuery1.sql - ch...CHARLOS\asus (69)* # X
  WHERE
    Kuantitas > 2;

/*11. Tampilkan Id_Produk dan Nama_Pesanan dari tabel Produk. Urutkan data berdasarkan id paling awal! */
SELECT
    P.ID_Produk,
    DP.Nama_Pesanan
FROM
    Produk P
LEFT JOIN
    Detail_Pesanan DP ON P.ID_Produk = DP.Id_Produk
ORDER BY
    P.ID_Produk ASC;
```

ID_Produk	Nama_Pesanan
11YS1	Baju
11YS2	Celana Pendek
11YS3	Sandal Gunung
11YS4	Sepatu
11YS5	Hoodie

99 %

Results Messages

charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 5 rows

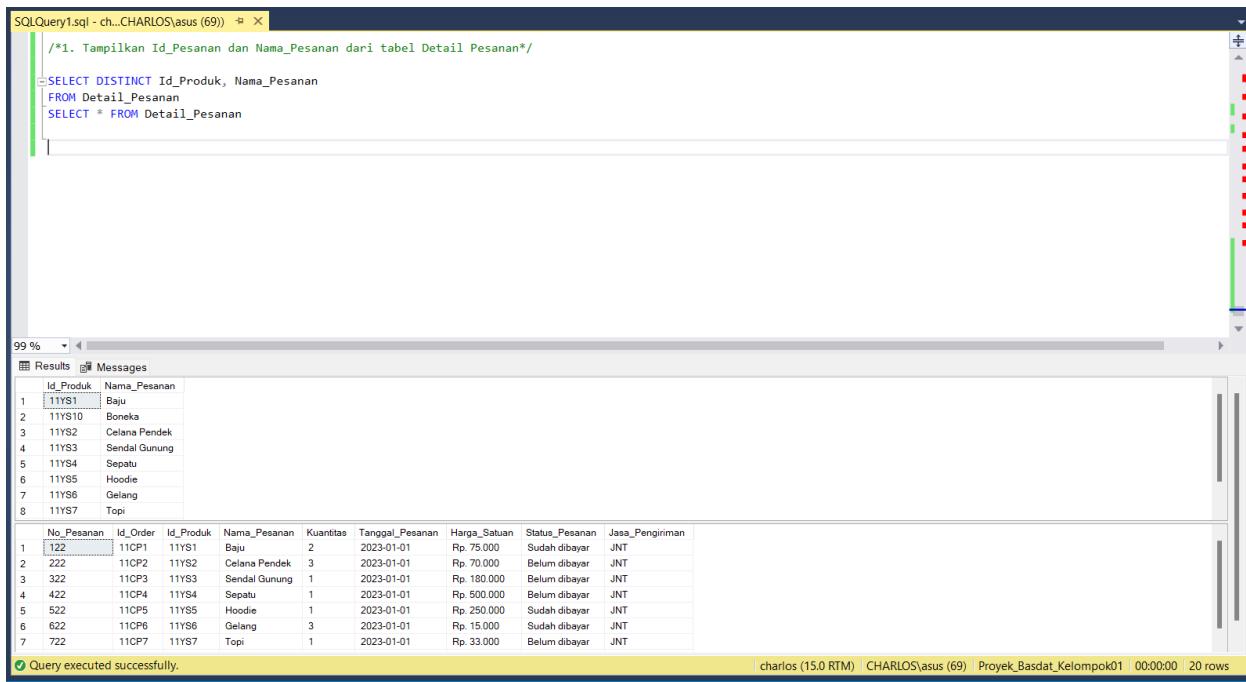
Query executed successfully.

Gambar 57. Menampilkan Id_Prodak dan Nama_Pesanan dari Tabel Produk

4.4 Quering Multiple Tabel

Syntax

```
/*1. Tampilkan Id_Pesanan dan Nama_Pesanan dari tabel Detail Pesanan*/  
  
SELECT DISTINCT Id_Produk, Nama_Pesanan  
FROM Detail_Pesanan  
SELECT * FROM Detail_Pesanan
```



The screenshot shows the SQL Query Editor window in SSMS. The query is displayed in the top pane:

```
/*1. Tampilkan Id_Pesanan dan Nama_Pesanan dari tabel Detail Pesanan*/  
  
SELECT DISTINCT Id_Produk, Nama_Pesanan  
FROM Detail_Pesanan  
SELECT * FROM Detail_Pesanan
```

The bottom pane shows the results of the query. It contains two tables: one for distinct products and one for all details.

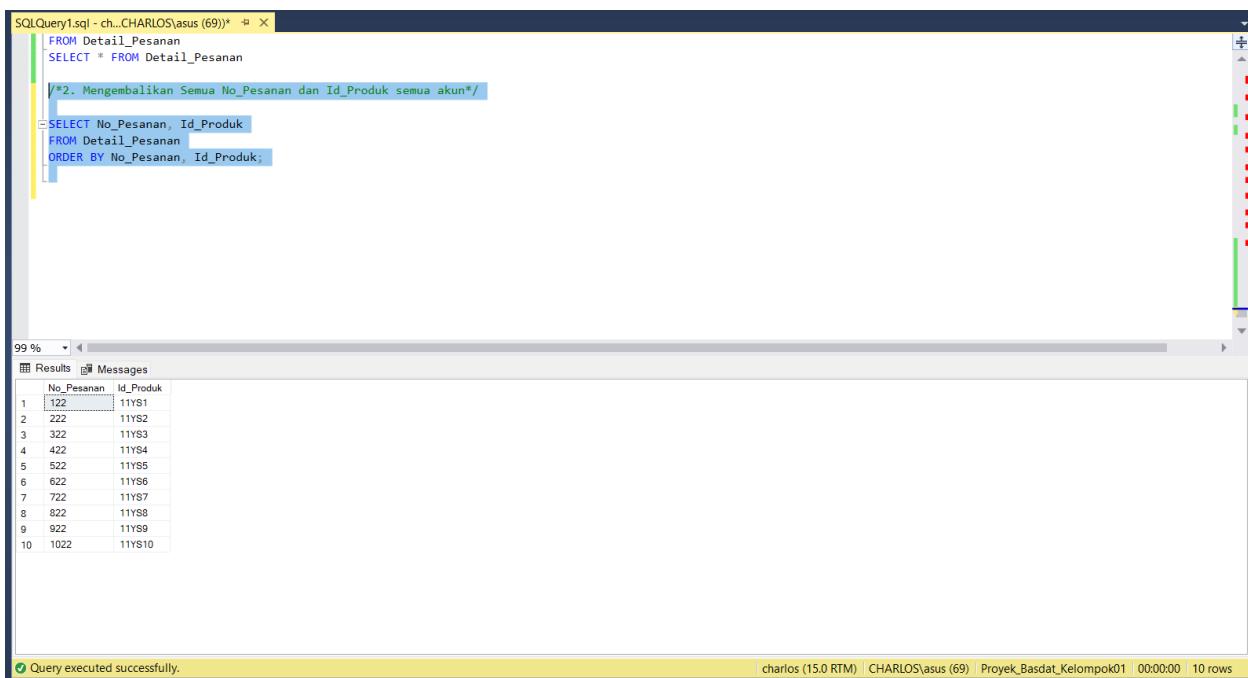
No	Id_Produk	Nama_Pesanan
1	11YS1	Baju
2	11YS10	Boneka
3	11YS2	Celana Pendek
4	11YS3	Sandal Gunung
5	11YS4	Sepatu
6	11YS5	Hoodie
7	11YS6	Gelang
8	11YS7	Topi

No_Pesanan	Id_Order	Id_Produk	Nama_Pesanan	Kuantitas	Tanggal_Pesanan	Harga_Satuan	Status_Pesanan	Jasa_Pengiriman
1	11CP1	11YS1	Baju	2	2023-01-01	Rp. 75.000	Sudah dibayar	JNT
2	222	11CP2	11YS2	3	2023-01-01	Rp. 70.000	Belum dibayar	JNT
3	322	11CP3	11YS3	1	2023-01-01	Rp. 180.000	Belum dibayar	JNT
4	422	11CP4	11YS4	1	2023-01-01	Rp. 500.000	Belum dibayar	JNT
5	522	11CP5	11YS5	1	2023-01-01	Rp. 250.000	Sudah dibayar	JNT
6	622	11CP6	11YS6	3	2023-01-01	Rp. 15.000	Sudah dibayar	JNT
7	722	11CP7	11YS7	1	2023-01-01	Rp. 33.000	Belum dibayar	JNT

Gambar 58. Show Different Value dari Id_Produk dan Nama_Pesanan

Syntax

```
/*2. Mengembalikan Semua No_Pesanan dan Id_Produk semua akun*/  
  
SELECT No_Pesanan, Id_Produk  
FROM Detail_Pesanan  
ORDER BY No_Pesanan, Id_Produk;
```



The screenshot shows the SQL Server Management Studio interface. In the top pane, there is a code editor window titled "SQLQuery1.sql - ch...CHARLOS(asus (69))". It contains the following SQL code:

```
FROM Detail_Pesanan  
SELECT * FROM Detail_Pesanan  
  
/*2. Mengembalikan Semua No_Pesanan dan Id_Produk semua akun*/  
  
SELECT No_Pesanan, Id_Produk  
FROM Detail_Pesanan  
ORDER BY No_Pesanan, Id_Produk;
```

In the bottom pane, there is a results grid titled "Results". The data is as follows:

No_Pesanan	Id_Produk
1	122
2	222
3	322
4	422
5	522
6	622
7	722
8	822
9	922
10	1022

At the bottom of the results grid, there is a status bar with the message "Query executed successfully." and other system information.

Gambar 59. Show All No_Pesanan dan d_Produk dari Setiap Pemesanan yang dilakukan user

Syntax

```
/* 3. Menggunakan Kombinasi Nilai di No_Pesanan dan Nama_Pesanan untuk Mengevaluasi  
Duplikat */  
  
SELECT DISTINCT No_Pesanan, Nama_Pesanan  
FROM Detail_Pesanan
```

The screenshot shows a SQL Server Management Studio (SSMS) interface. In the top pane, there is a code editor with the following SQL query:

```
/* 3. Menggunakan Kombinasi Nilai di No_Pesanan dan Nama_Pesanan untuk Mengevaluasi Duplikat */
SELECT DISTINCT No_Pesanan, Nama_Pesanan
FROM Detail_Pesanan
```

In the bottom pane, the results of the query are displayed in a table:

No_Pesanan	Nama_Pesanan
1 122	Baju
2 222	Celana Pendek
3 322	Sandal Gunung
4 422	Sepatu
5 522	Hoodie
6 622	Gelang
7 722	Topi
8 822	Jam
9 922	Kaca mata
10 1022	Boneka

A status bar at the bottom indicates: "Query executed successfully." and "charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 10 rows".

Gambar 60. Show All Bagian yang berbeda dari semua Pesanan

Syntax

```
/* 4. Featuring Nama_Pesanan along with Pesanan material*/

SELECT DISTINCT Nama_Pesanan
FROM Detail_Pesanan
GROUP BY Nama_Pesanan
```

The screenshot shows a SQL Server Management Studio (SSMS) interface. In the top-left corner, there's a tab labeled "SQLQuery1.sql - ch...CHARLOS\asus (69)*". Below the tabs, the main area contains a SQL query:

```
/* 4. Featuring Nama_Pesanan along with Pesanan material*/
SELECT DISTINCT Nama_Pesanan
FROM Detail_Pesanan
GROUP BY Nama_Pesanan
```

Below the query, the results pane displays a table titled "Results" with one column "Nama_Pesanan" containing 10 unique items:

Nama_Pesanan
1 Baju
2 Boneka
3 Celana Pendek
4 Gelang
5 Hoodie
6 Jam
7 Kaca mata
8 Sendal Gunung
9 Sepatu
10 Topi

At the bottom of the results pane, a status bar indicates: "Query executed successfully." followed by session details: "charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 10 rows".

Gambar 61. Pengelompokan Data Menjadi Satu Data yang unik dari Tabel Detail Pesanan

Syntax

```
/* 5. Count Number of Unique Rows */

SELECT COUNT(DISTINCT Nama_Pesanan)
FROM Detail_Pesanan
```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS\asus (69)*". The query is:

```
/* 5. Count Number of Unique Rows */
SELECT COUNT(DISTINCT Nama_Pesanan)
FROM Detail_Pesanan
```

The results pane shows a single row with the value 10. The status bar at the bottom indicates "Query executed successfully." and other session details.

Gambar 62. Count Number Of Unique Rows dari Tabel Detail Pesanan

4.5 Nested Sub-Queries

Syntax

```
/*1. Menampilkan Nama dan No_telepon dari Akun yang memiliki Email 'gmail.com'*/
SELECT Nama, No_telepon
FROM Akun
WHERE Email IN (
    SELECT Email
    FROM Akun
    WHERE Email LIKE '%@gmail.com'
);
```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS\asus (69)*". The query is:

```

/*1. Menampilkan Nama dan No_telepon dari Akun yang memiliki Email 'gmail.com'*/
SELECT Nama, No_telepon
FROM Akun
WHERE Email IN (
    SELECT Email
    FROM Akun
    WHERE Email LIKE '%@gmail.com'
)

```

The results window displays the following data:

	Nama	No_telepon
1	Charles	0812-1809-9848
2	Yessi	0812-6328-0007
3	Era	0858-3105-7136
4	Kartika	0813-6154-1794
5	Christian	0812-6013-6699
6	Chalvin	0853-6520-5722
7	Anton	0822-6021-5832
8	Lena	0812-7723-5987
9	Dilan	0813-8012-1714
10	Daniel	0853-9654-0004

At the bottom of the results window, it says "Query executed successfully." and shows the session details: charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 10 rows.

Gambar 63. Menampilkan Nama dan No_telepon dari Akun yang memiliki Email 'gmail.com'

Syntax

```

/*2. Menampilkan Nama dari Akun yang memiliki Id_Akun unik (tidak ada yang sama dengan akun lain)*/
SELECT Nama
FROM Akun
WHERE (SELECT COUNT(*) FROM Akun AS A WHERE A.Id_Akun = Akun.Id_Akun) = 1;

```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS\asus (69)*". The query is:

```
/*2. Menampilkan Nama dari Akun yang memiliki Id_Akun unik (tidak ada yang sama dengan akun lain)*/
SELECT Nama
FROM Akun
WHERE (SELECT COUNT(*) FROM Akun AS A WHERE A.Id_Akun = Akun.Id_Akun) = 1;
```

The results window shows a table with one column "Nama" containing 10 rows of names:

	Nama
1	Charles
2	Yessi
3	Era
4	Kartika
5	Christian
6	Chalvin
7	Anton
8	Lena
9	Dilan
10	Daniel

At the bottom of the results window, it says "Query executed successfully." and "charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 10 rows".

Gambar 64. Menampilkan Nama dari Akun yang memiliki Id_Akun unik (tidak ada yang sama dengan akun lain)

Syntax

```
/*3. Menampilkan Id_Produk dan Nama_Pesanan dari pesanan yang Status_Pesanan-nya sudah dibayar ('Sudah dibayar')/
```

```
SELECT Id_Produk, Nama_Pesanan
FROM Detail_Pesanan
WHERE No_Pesanan IN (
    SELECT No_Pesanan
    FROM Detail_Pesanan
    WHERE Status_Pesanan = 'Sudah dibayar'
);
```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS\asus (69)". The query is:

```
/*3. Menampilkan Id_Produk dan Nama_Pesanan dari pesanan yang Status_Pesanan-nya sudah dibayar ('Sudah dibayar')*/
SELECT Id_Produk, Nama_Pesanan
FROM Detail_Pesanan
WHERE No_Pesanan IN (
    SELECT No_Pesanan
    FROM Detail_Pesanan
    WHERE Status_Pesanan = 'Sudah dibayar'
);
```

The results pane shows a table with four rows:

	Id_Produk	Nama_Pesanan
1	11YS1	Baju
2	11YS5	Hoodie
3	11YS6	Gelang
4	11YS10	Boneka

At the bottom, a message bar indicates: "Query executed successfully." and shows connection details: charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok0 | 00:00:00 | 4 rows.

Gambar 65. Menampilkan Id_Produk dan Nama_Pesanan dari pesanan yang Status_Pesanan-nya sudah dibayar ('Sudah dibayar')

Syntax

```
/*4. Menampilkan No_Pesanan dan Tanggal_Pesanan dari pesanan dengan Kuantitas terbanyak*/
SELECT No_Pesanan, Tanggal_Pesanan
FROM Detail_Pesanan
WHERE Kuantitas = (
    SELECT MAX(Kuantitas)
    FROM Detail_Pesanan
);
```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS\asus (69)*". The query is:

```
/*4. Menampilkan No_Pesanan dan Tanggal_Pesanan dari pesanan dengan Kuantitas terbanyak*/
SELECT No_Pesanan, Tanggal_Pesanan
FROM Detail_Pesanan
WHERE Kuantitas = (
    SELECT MAX(Kuantitas)
    FROM Detail_Pesanan
);
```

The results pane shows a table with two rows:

	No_Pesanan	Tanggal_Pesanan
1	222	2023-01-01
2	622	2023-01-01

At the bottom, a message bar indicates: "Query executed successfully." and "charlos (15.0 RTM) CHARLOS\asus (69) Proyek_Basdat_Kelompok01 00:00:00 2 rows".

Gambar 66. Menampilkan No_Pesanan dan Tanggal_Pesanan dari pesanan dengan Kuantitas terbanyak

Syntax

```
/*5. Menampilkan No_Pesanan dan Id_Order dari pesanan yang menggunakan Jasa_Pengiriman 'JNT' dan belum dibayar*/

SELECT No_Pesanan, Id_Order
FROM Detail_Pesanan
WHERE Status_Pesanan = 'Belum dibayar'
AND Jasa_Pengiriman = 'JNT';
```

```

SQLQuery1.sql - ch...CHARLOS(asus (69)) ×
/*
5. Menampilkan No_Pesanan dan Id_Order dari pesanan yang menggunakan Jasa_Pengiriman 'JNT' dan belum dibayar*/
SELECT No_Pesanan, Id_Order
FROM Detail_Pesanan
WHERE Status_Pesanan = 'Belum dibayar'
AND Jasa_Pengiriman = 'JNT';

```

Results

No_Pesanan	Id_Order
1 222	11CP2
2 322	11CP3
3 422	11CP4
4 722	11CP7
5 822	11CP8
6 922	11CP9

Query executed successfully.

Gambar 67. Menampilkan No_Pesanan dan Id_Order dari pesanan yang menggunakan Jasa_Pengiriman 'JNT' dan belum dibayar

4.6 Join Tabel

Syntax

```

/* 1. INNER JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan
ID_Pelanggan */

SELECT Pelanggan.Id_Pelanggan, Pelanggan.Nama AS Nama_Pelanggan, Pelanggan.Alamat AS
Alamat_Pelanggan,
      Penjual.ID_Penjual, Penjual>Nama_Toko, Penjual.Alamat_Toko
FROM Pelanggan
INNER JOIN Penjual ON Pelanggan.Id_Pelanggan = Penjual.ID_Penjual;

```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS(asus (69))". The query performs an INNER JOIN between the "Pelanggan" and "Penjual" tables based on their "ID_Pelanggan" and "ID_Penjual" respectively. The results are displayed in a table with columns: Id_Pelanggan, Nama_Pelanggan, Alamat_Pelanggan, ID_Penjual, Name_Toko, and Alamat_Toko. The data shows three rows of information.

	Id_Pelanggan	Nama_Pelanggan	Alamat_Pelanggan	ID_Penjual	Name_Toko	Alamat_Toko
1	11S22	Lena	Medan	11S22	Charles Shop	Jakarta Timur
2	13S22	Andika	Medan	13S22	Mari Belanja	Medan
3	14S22	Yessi	Tarutung	14S22	Jamet Store	Kalimantan

At the bottom of the window, a message bar indicates: "Query executed successfully." and shows the session details: charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 3 rows.

Gambar 68. INNER JOIN - Menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan

Syntax

```
/* 2. LEFT JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan
ID_Pelanggan */

SELECT *
FROM Pelanggan P
LEFT JOIN Penjual PJ ON P.Id_Pelanggan = PJ.ID_Penjual;
```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS(asus (69))". The query is:

```

/* 2. LEFT JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan */

SELECT *
FROM Pelanggan P
LEFT JOIN Penjual PJ ON P.Id_Pelanggan = PJ.ID_Penjual;

```

The results grid displays 10 rows of data from the joined tables:

ID_Pelanggan	Nama	Alamat	ID_Penjual	Nama_Toko	Alamat_Toko
11822	Lena	Medan	11S22	Charles Shop	Jakarta Timur
12522	Dilan	Sibolga	NULL	NULL	NULL
13522	Andika	Medan	13S22	Man Belanja	Medan
14522	Yessi	Tanutung	14S22	Janet Store	Kalimantan
16522	Anton	Balige	NULL	NULL	NULL
21922	David	Medan	NULL	NULL	NULL
22522	Andre	Tanutung	NULL	NULL	NULL
23522	Tian	Balige	NULL	NULL	NULL
24522	Johan	Medan	NULL	NULL	NULL
25522	Mitha	Sibolga	NULL	NULL	NULL

At the bottom, a message bar indicates: "Query executed successfully." and shows connection details: charlos (15.0 RTM) | CHARLOS(asus (69)) | Proyek_Basdat_Kelompok01 | 00:00:00 | 10 rows.

Gambar 69. LEFT JOIN - Menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan

Syntax

```

/* 3. RIGHT JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan
ID_Pelanggan */

SELECT *
FROM Penjual
RIGHT JOIN Pelanggan ON Penjual.ID_Penjual = Pelanggan.Id_Pelanggan;

```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS\asus (69)". The query is:

```

/* 3. RIGHT JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan */

SELECT *
FROM Penjual
RIGHT JOIN Pelanggan ON Penjual.ID_Penjual = Pelanggan.ID_Pelanggan;

```

The results grid displays 10 rows of data from the joined tables:

ID_Penjual	Nama_Toko	Alamat_Toko	Id_Pelanggan	Nama	Alamat
1 11S22	Charles Shop	Jakarta Timur	11S22	Lena	Medan
2 NULL	NULL	NULL	12S22	Dilan	Sibolga
3 13S22	Mari Belanja	Medan	13S22	Andika	Medan
4 14S22	Jamet Store	Kalimantan	14S22	Yessi	Tarutung
5 NULL	NULL	NULL	16S22	Anton	Balige
6 NULL	NULL	NULL	21S22	David	Medan
7 NULL	NULL	NULL	22S22	Andre	Tarutung
8 NULL	NULL	NULL	23S22	Tian	Balige
9 NULL	NULL	NULL	24S22	Johan	Medan
10 NULL	NULL	NULL	25S22	Mitha	Sibolga

At the bottom, a message bar indicates: "Query executed successfully." and shows session details: charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 10 rows.

Gambar 70. RIGHT JOIN - Menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan

Syntax

```

/* 4. OUTER JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan
ID_Pelanggan */

SELECT Pelanggan.Id_Pelanggan, Pelanggan.Nama AS Nama_Pelanggan, Pelanggan.Alamat AS
Alamat_Pelanggan, Penjual.Nama_Toko, Penjual.Alamat_Toko
FROM Pelanggan
FULL OUTER JOIN Penjual ON Pelanggan.Id_Pelanggan = Penjual.ID_Penjual;

```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS\asus (69)". The query is:

```

/* 4. OUTER JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan */
SELECT Pelanggan.Id_Pelanggan, Pelanggan.Nama AS Nama_Pelanggan, Pelanggan.Alamat AS Alamat_Pelanggan, Penjual.Nama_Toko, Penjual.Alamat_Toko
FROM Pelanggan
    FULL OUTER JOIN Penjual ON Pelanggan.Id_Pelanggan = Penjual.ID_Penjual;

```

The results window displays the following data:

	Id_Pelanggan	Nama_Pelanggan	Alamat_Pelanggan	Nama_Toko	Alamat_Toko
1	11922	Lena	Medan	Charles Shop	Jakarta Timur
2	12522	Dilan	Sibolga	NULL	NULL
3	13522	Andika	Medan	Mari Belanja	Medan
4	14522	Yessi	Tarutung	Jamet Store	Kalimantan
5	NULL	NULL	NULL	Serba ada	Medan
6	16522	Anton	Balige	NULL	NULL
7	NULL	NULL	NULL	Your Fashion	Jakarta Timur
8	21522	David	Medan	NULL	NULL
9	22522	Andre	Tarutung	NULL	NULL
10	23522	Tian	Balige	NULL	NULL
11	24522	Johan	Medan	NULL	NULL
12	25522	Mitha	Sibolga	NULL	NULL

At the bottom, a message bar indicates: "Query executed successfully." and shows the session details: charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 12 rows.

Gambar 71. OUTER JOIN - Menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan

Syntax

```

/* 5. CROSS JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan */

SELECT p.Id_Pelanggan, p.Nama AS 'Nama Pelanggan', p.Alamat AS 'Alamat Pelanggan',
pj.ID_Penjual, pj.Nama_Toko AS 'Nama Toko', pj.Alamat_Toko AS 'Alamat Toko'
FROM Pelanggan p
CROSS JOIN Penjual pj;

```

```

SQLQuery1.sql - ch...CHARLOS\asus (69)  ↗ X

/*
  5. CROSS JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan */
SELECT p.Id_Pelanggan, p.Nama AS 'Nama Pelanggan', p.Alamat AS 'Alamat Pelanggan',
pj.ID_Penjual, pj>Nama_Toko AS 'Nama Toko', pjAlamat_Toko AS 'Alamat Toko'
FROM Pelanggan p
CROSS JOIN Penjual pj;

```

Results Messages

ID_Pelanggan	Nama Pelanggan	Alamat Pelanggan	ID_Penjual	Nama Toko	Alamat Toko
1 11S22	Lena	Medan	11S22	Charles Shop	Jakarta Timur
2 12S22	Dilan	Sibolga	11S22	Charles Shop	Jakarta Timur
3 13S22	Andika	Medan	11S22	Charles Shop	Jakarta Timur
4 14S22	Yessi	Tarutung	11S22	Charles Shop	Jakarta Timur
5 16S22	Anton	Balige	11S22	Charles Shop	Jakarta Timur
6 21S22	David	Medan	11S22	Charles Shop	Jakarta Timur
7 22S22	Andre	Tarutung	11S22	Charles Shop	Jakarta Timur
8 23S22	Tian	Balige	11S22	Charles Shop	Jakarta Timur
9 24S22	Johan	Medan	11S22	Charles Shop	Jakarta Timur
10 25S22	Mitha	Sibolga	11S22	Charles Shop	Jakarta Timur
11 1S22	Lena	Medan	13S22	Mari Belanja	Medan
12 12S22	Dilan	Sibolga	13S22	Mari Belanja	Medan
13 13S22	Andika	Medan	13S22	Mari Belanja	Medan
14 14S22	Yessi	Tarutung	13S22	Mari Belanja	Medan
15 16S22	Anton	Balige	13S22	Mari Belanja	Medan
16 21S22	David	Medan	13S22	Mari Belanja	Medan
17 22S22	Andre	Tarutung	13S22	Mari Belanja	Medan
18 23S22	Tian	Balige	13S22	Mari Belanja	Medan
19 24S22	Johan	Medan	13S22	Mari Belanja	Medan
20 25S22	Mitha	Sibolga	13S22	Mari Belanja	Medan
21 26S22	Lena	Medan	13S22	Mari Belanja	Medan
22 27S22	Dilan	Sibolga	13S22	Mari Belanja	Medan
23 28S22	Andika	Medan	13S22	Mari Belanja	Medan
24 29S22	Yessi	Tarutung	13S22	Mari Belanja	Medan
25 30S22	Anton	Balige	13S22	Mari Belanja	Medan
26 31S22	David	Medan	13S22	Mari Belanja	Medan
27 32S22	Andre	Tarutung	13S22	Mari Belanja	Medan
28 33S22	Tian	Balige	13S22	Mari Belanja	Medan
29 34S22	Johan	Medan	13S22	Mari Belanja	Medan
30 35S22	Mitha	Sibolga	13S22	Mari Belanja	Medan
31 36S22	Lena	Medan	13S22	Mari Belanja	Medan
32 37S22	Dilan	Sibolga	13S22	Mari Belanja	Medan
33 38S22	Andika	Medan	13S22	Mari Belanja	Medan
34 39S22	Yessi	Tarutung	13S22	Mari Belanja	Medan
35 40S22	Anton	Balige	13S22	Mari Belanja	Medan
36 41S22	David	Medan	13S22	Mari Belanja	Medan
37 42S22	Andre	Tarutung	13S22	Mari Belanja	Medan
38 43S22	Tian	Balige	13S22	Mari Belanja	Medan
39 44S22	Johan	Medan	13S22	Mari Belanja	Medan
40 45S22	Mitha	Sibolga	13S22	Mari Belanja	Medan
41 46S22	Lena	Medan	13S22	Mari Belanja	Medan
42 47S22	Dilan	Sibolga	13S22	Mari Belanja	Medan
43 48S22	Andika	Medan	13S22	Mari Belanja	Medan
44 49S22	Yessi	Tarutung	13S22	Mari Belanja	Medan
45 50S22	Anton	Balige	13S22	Mari Belanja	Medan
46 51S22	David	Medan	13S22	Mari Belanja	Medan
47 52S22	Andre	Tarutung	13S22	Mari Belanja	Medan
48 53S22	Tian	Balige	13S22	Mari Belanja	Medan
49 54S22	Johan	Medan	13S22	Mari Belanja	Medan
50 55S22	Mitha	Sibolga	13S22	Mari Belanja	Medan

Query executed successfully.

Gambar 72. CROSS JOIN - Menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan ID_Pelanggan

4.7 Creating Views

Syntax

```

/*
  1. View 1 - Daftar Detail Pesanan */

CREATE VIEW Daftar_Pesanan AS
SELECT No_Pesanan, Nama_Pesanan, Kuantitas, Tanggal_Pesanan, Harga_Satuan
FROM Detail_Pesanan;

SELECT * FROM Daftar_Pesanan;

```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS\asus (69)". The code creates a view named "Daftar_Pesanan" with the following definition:

```
/* 1. View 1 - Daftar Detail Pesanan */
CREATE VIEW Daftar_Pesanan AS
SELECT No_Pesanan, Nama_Pesanan, Kuantitas, Tanggal_Pesanan, Harga_Satuan
FROM Detail_Pesanan;
SELECT * FROM Daftar_Pesanan;
```

Below the code, the results pane displays a table with 10 rows of data from the "Detail_Pesanan" table:

No_Pesanan	Nama_Pesanan	Kuantitas	Tanggal_Pesanan	Harga_Satuan
1 122	Baju	2	2023-01-01	Rp. 75.000
2 222	Celana Pendek	3	2023-01-01	Rp. 70.000
3 322	Sandal Gunung	1	2023-01-01	Rp. 180.000
4 422	Sepatu	1	2023-01-01	Rp. 500.000
5 522	Hoodie	1	2023-01-01	Rp. 250.000
6 622	Gelang	3	2023-01-01	Rp. 15.000
7 722	Topi	1	2023-01-01	Rp. 33.000
8 822	Jam	1	2023-01-01	Rp. 500.000
9 922	Kaca mata	1	2023-01-01	Rp. 500.000
10 1022	Boneka	1	2023-01-01	Rp. 300.000

At the bottom of the results pane, a message indicates: "Query executed successfully." and shows the session details: "charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 10 rows".

Gambar 73. View 1 - Daftar Detail Pesanan

Syntax

```
/* 2. View 2 - Daftar Pelanggan */

CREATE VIEW Daftar_Pelanggan AS
SELECT Id_Pelanggan, Nama, Alamat
FROM Pelanggan;

SELECT * FROM Daftar_Pelanggan;
```

The screenshot shows a SQL query window titled 'SQLQuery1.sql - ch...CHARLOS\asus (69)'. The code in the query pane is:

```

/* 2. View 2 - Daftar Pelanggan */
CREATE VIEW Daftar_Pelanggan AS
SELECT Id_Pelanggan, Nama, Alamat
FROM Pelanggan;
SELECT * FROM Daftar_Pelanggan;

```

The results pane displays a table with 10 rows of data:

	Id_Pelanggan	Nama	Alamat
1	11S22	Lena	Medan
2	12S22	Dilan	Medan
3	13S22	Andika	Medan
4	14S22	Yessi	Tarutung
5	16S22	Anton	Balige
6	21S22	David	Medan
7	22S22	Andre	Tarutung
8	23S22	Tian	Balige
9	24S22	Johan	Medan
10	25S22	Mitha	Sibolga

At the bottom of the results pane, a message indicates: 'Query executed successfully.' and shows the session details: 'charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 10 rows'.

Gambar 74. View 2 - Daftar Pelanggan

Syntax

```

/* 3. View 3 - Detail Order Pelanggan */

CREATE VIEW Order_Pelanggan AS
SELECT O.Id_Order, O.Nama_Produk, O.Harga, O.Kuantitas, P.Nama AS 'Nama Pelanggan'
FROM Orders O
JOIN Pelanggan P ON O.Id_Pelanggan = P.Id_Pelanggan;

SELECT * FROM Order_Pelanggan;

```

The screenshot shows a SQL query window titled 'SQLQuery1.sql - ch...CHARLOS\asus (69)'. The query creates a view named 'Order_Pelanggan' that joins the 'Orders' table with the 'Pelanggan' table. The results window shows 10 rows of data, each containing an order ID, product name, price, quantity, and customer name.

```

/*
3. View 3 - Detail Order Pelanggan */

CREATE VIEW Order_Pelanggan AS
SELECT O.Id_Order, O.Nama_Produk, O.Harga, O.Kuantitas, P.Nama AS 'Nama Pelanggan'
FROM Orders O
JOIN Pelanggan P ON O.Id_Pelanggan = P.Id_Pelanggan;

SELECT * FROM Order_Pelanggan;

```

	Id_Order	Nama_Produk	Harga	Kuantitas	Nama Pelanggan
1	11CP1	Baju	Rp.75.000	2	Andika
2	11CP10	Boneka	Rp.300.000	1	Mitha
3	11CP2	Celana Pendek	Rp.70.000	3	Yessi
4	11CP3	Sandal Gungung	Rp.180.000	1	Anton
5	11CP4	Sepatu	Rp.500.000	1	Lena
6	11CP5	Hoodie	Rp.250.000	1	Dilan
7	11CP6	Gelang	Rp.15.000	3	David
8	11CP7	Topi	Rp.33.000	1	Andre
9	11CP8	Jam	Rp.500.000	1	Tian
10	11CP9	Kaca mata	Rp.500.000	1	Johan

Query executed successfully. charlos (15.0 RTM) | CHARLOS\asus (69) | Proyek_Basdat_Kelompok01 | 00:00:00 | 10 rows

Gambar 75. View 3 - Detail Order Pelanggan

Syntax

```

/*
4. View 4 - Tinjauan Produk */

CREATE VIEW Tinjauan_Produk AS
SELECT T.ID_Tinjauan, T.Isi_Tinjauan, T.Tanggal_Tinjauan, T.Rating, P.Nama_Produk
FROM Tinjauan T
JOIN Produk P ON T.ID_Produk = P.ID_Produk;

SELECT * FROM Tinjauan_Produk;

```

```

SQLQuery1.sql - ch...CHARLOS\asus (69) × X

/* 4. View 4 - Tinjauan Produk */

CREATE VIEW Tinjauan_Produk AS
SELECT T.ID_Tinjauan, T.Isi_Tinjauan, T.Tanggal_Tinjauan, T.Rating, P.Nama_Produk
FROM Tinjauan T
JOIN Produk P ON T.ID_Produk = P.ID_Produk;

SELECT * FROM Tinjauan_Produk;

```

Results Messages

ID_Tinjauan	Isi_Tinjauan	Tanggal_Tinjauan	Rating	Nama_Produk
001T1	Kualitas barang baik	2023-01-06	4	Baju
001T2	Kualitas barang sangat baik	2023-01-07	5	Celana Pendek
001T3	Kualitas barang baik	2023-01-08	3	Sandal Gunung
001T4	Kualitas barang kurang baik	2023-01-09	2	Sepatu
001T5	Kualitas barang baik	2023-01-10	4	Hoodie

Query executed successfully. charlos (15.0 RTM) CHARLOS\asus (69) Proyek_Basdat_Kelompok01 00:00:00 5 rows

Gambar 76. View 4 - Tinjauan Produk

Syntax

```

/* 5. View 5 - Penjual dan Pengiriman */

CREATE VIEW Penjual_Pengiriman AS
SELECT PJ.ID_Penjual, PJ.Nama_Toko, PG.No_Resi, PG.Tanggal_Pesanan, PG.Status_Pengiriman
FROM Penjual PJ
JOIN Pengiriman PG ON PJ.ID_Penjual = PG.ID_Penjual;

SELECT * FROM Penjual_Pengiriman;

```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS\asus (69)". The query creates a view named "Penjual_Pengiriman" which joins two tables: "Penjual" and "Pengiriman". The result set displays five rows of data.

```
/* 5. View 5 - Penjual dan Pengiriman */
CREATE VIEW Penjual_Pengiriman AS
SELECT PJ.ID_Penjual, PJ.Nama_Toko, PG.No_Resi, PG.Tanggal_Pesanan, PG.Status_Pengiriman
FROM Penjual PJ
JOIN Pengiriman PG ON PJ.ID_Penjual = PG.ID_Penjual;

SELECT * FROM Penjual_Pengiriman;
```

ID_Penjual	Nama_Toko	No_Resi	Tanggal_Pesanan	Status_Pengiriman
1 11S22	Carlos Shop	JNT01	2023-01-01	Dikemas
2 13S22	Mari Belanja	JNT02	2023-01-01	Dalam Perjalanan
3 14S22	Janet Store	JNT03	2023-01-01	Dalam Perjalanan
4 15S22	Serba ada	JNT04	2023-01-01	Dikemas
5 20S22	Your Fashion	JNT05	2023-01-01	Dikemas

Query executed successfully.

Gambar 77. View 5 - Penjual dan Pengiriman

4.8 Function

Syntax

```
-- Menghitung total harga pesanan
CREATE FUNCTION HitungTotalHargaFormatted(@HargaSatuan VARCHAR(20), @Kuantitas INT)
RETURNS VARCHAR(20)
AS
BEGIN
    DECLARE @Total INT
    SET @Total = CAST(REPLACE(REPLACE(@HargaSatuan, 'Rp. ', ''), '.', '') AS INT) *
@Kuantitas
    RETURN 'Rp.' + FORMAT(@Total, 'N0')
END

SELECT dbo.HitungTotalHargaFormatted(Harga_Satuan, Kuantitas) AS Total_Harga_Formatted
FROM Detail_Pesanan;
```

The screenshot shows a SQL query window titled "SQLQuery1.sql - ch...CHARLOS\asus (55)". The query itself is:

```
-- Membuat fungsi untuk menghitung total harga pesanan
CREATE FUNCTION HitungTotalHargaFormatted(@HargaSatuan VARCHAR(20), @Kuantitas INT)
RETURNS VARCHAR(20)
AS
BEGIN
    DECLARE @Total INT
    SET @Total = CAST(REPLACE(REPLACE(@HargaSatuan, 'Rp. ', ''), '.', '') AS INT) * @Kuantitas
    RETURN 'Rp.' + FORMAT(@Total, 'N0')
END

SELECT dbo.HitungTotalHargaFormatted(Harga_Satuan, Kuantitas) AS Total_Harga_Formatted
FROM Detail_Pesanan;
```

The results pane shows a table with one column "Total_Harga_Formated". The data is as follows:

Total_Harga_Formated
1 Rp.150.000
2 Rp.210.000
3 Rp.180.000
4 Rp.500.000
5 Rp.250.000
6 Rp.45.000
7 Rp.33.000
8 Rp.500.000
9 Rp.500.000
10 Rp.300.000

At the bottom of the results pane, it says "Query executed successfully." and "10 rows".

Gambar 78. Menghitung Total Harga Pesanan

Syntax

```
-- Menghitung rata-rata harga dari produk

CREATE FUNCTION dbo.fn_HargaRataRata()
RETURNS DECIMAL(18, 2)
AS
BEGIN
    DECLARE @AvgPrice DECIMAL(18, 2);
    -- Query untuk menghitung rata-rata harga
    SELECT @AvgPrice = AVG(CAST(REPLACE(REPLACE(Harga, 'Rp. ', ''), '.', '') AS
DECIMAL(18, 2)))
    FROM Produk;
    RETURN @AvgPrice;
END;
GO

SELECT FORMAT(dbo.fn_HargaRataRata(), 'Rp ###,###,###.00') AS 'Rata-rata Harga';
```

The screenshot shows the SSMS interface with a query window titled 'SQLQuery1.sql - ch...CHARLOS\asus (55)*'. The code in the window is identical to the one above. Below the code, the 'Results' tab is selected, displaying a single row of results:

Rata-rata Harga
1 Rp 215.000,00

At the bottom of the window, a status bar indicates: 'Query executed successfully.' and 'charlos (15.0 RTM) | CHARLOS\asus (55) | Proyek_Basdat_Kelompok01 | 00:00:00 | 1 rows'.

Gambar 79. Menghitung Rata-rata Harga dari Produk

4.9 Procedure

Syntax

```
-- Menampilkan Semua Data dari Tabel Orders
CREATE PROCEDURE ShowOrders
AS
BEGIN
    SELECT * FROM Orders
END;
EXEC ShowOrders;
```

The screenshot shows the SQL Query Editor window in SSMS. The code area contains the stored procedure definition:

```
-- Menampilkan Semua Data dari Tabel Orders
CREATE PROCEDURE ShowOrders
AS
BEGIN
    SELECT * FROM Orders
END;
EXEC ShowOrders;
```

The 'Results' tab is selected, displaying the output of the executed query:

	Id_Order	Nama_Produk	Diskon	Harga	Kuantitas	Id_Pelanggan
1	11CP1	Baju	10%	Rp.75.000	2	13S22
2	11CP10	Boneka	10%	Rp.300.000	1	25S22
3	11CP2	Celana Pendek	20%	Rp.70.000	3	14S22
4	11CP3	Sandal Gunung	15%	Rp.180.000	1	16S22
5	11CP4	Sepatu	17%	Rp.500.000	1	11S22
6	11CP5	Hoodie	10%	Rp.250.000	1	12S22
7	11CP6	Gelang	10%	Rp.15.000	3	21S22
8	11CP7	Topi	20%	Rp.33.000	1	22S22
9	11CP8	Jam	15%	Rp.500.000	1	23S22
10	11CP9	Kaca mata	17%	Rp.500.000	1	24S22

At the bottom of the results pane, a message indicates the query was executed successfully.

Gambar 80. Menampilkan Seluruh Data dari Tabel Order

Syntax

```
-- Menampilkan seluruh data dari Tabel Detail Pesanan

CREATE PROCEDURE ShowOrderDetails
AS
BEGIN
    SELECT
        No_Pesanan,
        Id_Order,
        Id_Produk,
        Nama_Pesanan,
        Kuantitas,
        Tanggal_Pesanan,
        Harga_Satuan,
        Status_Pesanan,
        Jasa_Pengiriman
    FROM
        Detail_Pesanan;
END;

EXEC ShowOrderDetails;
```

The screenshot shows the SQL Query Editor window titled "SQLQuery1.sql - ch...CHARLOS(asus (55))". The query is a stored procedure named "ShowOrderDetails" that selects all columns from the "Detail_Pesanan" table. The results pane displays 10 rows of data.

No_Pesanan	Id_Order	Id_Produk	Nama_Pesanan	Kuantitas	Tanggal_Pesanan	Harga_Satuan	Status_Pesanan	Jasa_Pengiriman
1 122	11CP1	11YS1	Baju	2	2023-01-01	Rp. 75.000	Sudah dibayar	JNT
2 222	11CP2	11YS2	Celana Pendek	3	2023-01-01	Rp. 70.000	Belum dibayar	JNT
3 322	11CP3	11YS3	Sendal Gunung	1	2023-01-01	Rp. 180.000	Belum dibayar	JNT
4 422	11CP4	11YS4	Sepatu	1	2023-01-01	Rp. 500.000	Belum dibayar	JNT
5 522	11CP5	11YS5	Hoodie	1	2023-01-01	Rp. 250.000	Sudah dibayar	JNT
6 622	11CP6	11YS6	Gelang	3	2023-01-01	Rp. 15.000	Sudah dibayar	JNT
7 722	11CP7	11YS7	Topi	1	2023-01-01	Rp. 33.000	Belum dibayar	JNT
8 822	11CP8	11YS8	Jam	1	2023-01-01	Rp. 500.000	Belum dibayar	JNT
9 922	11CP9	11YS9	Kaca mata	1	2023-01-01	Rp. 500.000	Belum dibayar	JNT
10 1022	11CP10	11YS10	Boneka	1	2023-01-01	Rp. 300.000	Sudah dibayar	JNT

Gambar 81. Menampilkan Seluruh Data dari Tabel Detail Pesanan

BAB 5

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dalam konteks pengembangan sistem e-commerce, proses perancangan basis data melalui tahapan-tahapan seperti ERD, CDM, PDM, dan penggunaan kueri SQL sangat krusial. Tahap ini mencakup penggambaran model data, normalisasi, dan implementasi kueri untuk mendukung operasi dan fungsionalitas sistem.

Pemodelan basis data e-commerce mengintegrasikan informasi tentang pengguna, produk, pesanan, inventaris, pengiriman, dan banyak aspek lainnya. Ini memastikan hubungan yang baik antara entitas-entitas tersebut untuk mendukung proses transaksi, pelacakan pesanan, informasi produk, dan manajemen inventaris yang efisien.

Normalisasi berperan dalam memastikan struktur data yang optimal, mengurangi redudansi, dan mempertahankan konsistensi data. Proses ini membantu meningkatkan efisiensi penyimpanan data, meminimalkan kesalahan, dan memastikan integritas data.

Implementasi kueri SQL, termasuk penggunaan view, function, dan procedure, membantu dalam mengelola data dengan efektif. Dengan kueri yang tepat, sistem dapat mengakses, menyimpan, dan memanipulasi data sesuai kebutuhan, meningkatkan fungsionalitas sistem secara keseluruhan.

Dengan perancangan basis data yang kuat dan optimal, sistem e-commerce dapat menyediakan pengalaman pengguna yang baik, mendukung proses transaksi yang lancar, dan memberikan informasi yang relevan kepada pengguna.

5.2 Saran

Dengan dasar pengalaman dalam perancangan basis data sebelumnya, penting untuk memperhatikan batasan analisis dalam studi kasus berikutnya. Tujuannya adalah agar fokus tetap terjaga, mencegah penyimpangan yang berlebihan, serta memastikan pendekatan terhadap setiap entitas saling terhubung dengan baik dan relevan. Hal ini akan memungkinkan pemahaman yang mendalam terhadap topik yang dibahas.

DAFTAR PUSTAKA

Purnama, S., Hafizd, K. A., & Sayyidati, R. (2020). Sistem Informasi Kantin Elektronik (ECanteen) Politeknik Negeri Tanah Laut Berbasis Web Mobile. *Antivirus: Jurnal Ilmiah Teknik Informatika*, 14(2), 73-85.

Mariana, 2018. Sistem Informasi Permintaan Barang. *Teknik Informatika*. Politeknik Negeri Tanah Laut Pelaihari.

Syahrial, A, Akbar I. F, Anggraeni. L., Syahreza. R., Afifah. P. N. 2018. Perancangan Basis Data untuk Sistem Penjualan pada Kafet Neo.

Engel, "Conceptual Data Model dan Phisycal Data Model," Pap. Knowl. Towar. a Media Hist. Doc., 2018.

LAMPIRAN

```
-- CREATING AND MAINTING DATABASE
```

```
CREATE DATABASE Proyek_Basdat_Kelompok01
ON PRIMARY (NAME = EcommercePrimary,
            FILENAME = 'C:\Users\asus\Documents\E-commerce.mdf',
            SIZE = 10MB,
            MAXSIZE = 10000MB,
            FILEGROWTH = 20%),
            (NAME = EcommerceSecondary,
            FILENAME = 'C:\Users\asus\Documents\E-commerce.ndf',
            SIZE = 5MB,
            MAXSIZE = 10000MB,
            FILEGROWTH = 30%)
```

```
LOG ON
```

```
(NAME = EcommerceLog,
            FILENAME = 'C:\Users\asus\Documents\E-commerce.ldf',
            SIZE = 30MB,
            MAXSIZE = 5000MB,
            FILEGROWTH = 20%)
```

```
sp_helpdb Proyek_Basdat_Kelompok01
```

```
--Tabel Akun
```

```
CREATE TABLE Akun(
    Id_Akun varchar(10) PRIMARY KEY,
    Email varchar(50),
```

```
        Nama varchar(50),  
        No_Telepon varchar(50),  
    );  
  
INSERT INTO Akun (Id_Akun, Email, Nama, No_telepon)  
VALUES  
('11S22', 'charlos@gmail.com', 'Charlos', '0812-1809-9848'),  
('12S22', 'yessi@gmail.com', 'Yessi', '0812-6328-0007'),  
('13S22', 'esra@gmail.com', 'Esra', '0858-3105-7136'),  
('14S22', 'kartika@gmail.com', 'Kartika', '0813-6154-1794'),  
('15S22', 'christian@gmail.com', 'Christian', '0812-6013-6699'),  
('16S22', 'chalvin@gmail.com', 'Chalvin', '0853-6520-5722'),  
('17S22', 'anton@gmail.com', 'Anton', '0822-6021-5832'),  
('18S22', 'lena@gmail.com', 'Lena', '0812-7723-5987'),  
('19S22', 'dilan@gmail.com', 'Dilan', '0813-8012-1714'),  
('20S22', 'daniel@gmail.com', 'Daniel', '0853-9654-0004');
```

```
select * from Akun
```

```
-- Tabel Pelanggan
```

```
CREATE TABLE Pelanggan (  
    Id_Pelanggan varchar(10) PRIMARY KEY,  
    Nama varchar(50),  
    Alamat varchar(50),  
);
```

```
select * from Pelanggan
```

```
INSERT INTO Pelanggan (Id_Pelanggan, Nama, Alamat)
VALUES
('13S22', 'Chalvin', 'Medan'),
('14S22', 'Yessi', 'Tarutung'),
('16S22', 'Anton', 'Balige'),
('11S22', 'Lena', 'Medan'),
('12S22', 'Dilan', 'Sibolga');
```

```
INSERT INTO Pelanggan (Id_Pelanggan, Nama, Alamat)
VALUES
('21S22', 'David', 'Medan'),
('22S22', 'Andre', 'Tarutung'),
('23S22', 'Tian', 'Balige'),
('24S22', 'Johan', 'Medan'),
('25S22', 'Mitha', 'Sibolga');
```

```
select * from Pelanggan
```

```
-- Tabel Order
```

```
CREATE TABLE Orders (
    Id_Order varchar(10) PRIMARY KEY,
    Nama_Produk varchar(50),
    Diskon varchar(10),
    Harga varchar(10),
```

```

Kuantitas varchar(10),
Id_Pelanggan varchar(10) REFERENCES Pelanggan (Id_Pelanggan)
);

select * from Orders

INSERT INTO Orders (Id_Order, Nama_Produk, Diskon, Harga, Kuantitas, Id_Pelanggan)
VALUES
('11CP1', 'Baju', '10%', 'Rp.75.000', '2', '13S22'),
('11CP2', 'Celana Pendek', '20%', 'Rp.70.000', '3', '14S22'),
('11CP3', 'Sendal Gunung', '15%', 'Rp.180.000', '1', '16S22'),
('11CP4', 'Sepatu', '17%', 'Rp.500.000', '1', '11S22'),
('11CP5', 'Hoodie', '10%', 'Rp.250.000', '1', '12S22');

INSERT INTO Orders (Id_Order, Nama_Produk, Diskon, Harga, Kuantitas, Id_Pelanggan)
VALUES
('11CP6', 'Gelang', '10%', 'Rp.15.000', '3', '21S22'),
('11CP7', 'Topi', '20%', 'Rp.33.000', '1', '22S22'),
('11CP8', 'Jam', '15%', 'Rp.500.000', '1', '23S22'),
('11CP9', 'Kaca mata', '17%', 'Rp.500.000', '1', '24S22'),
('11CP10', 'Boneka', '10%', 'Rp.300.000', '1', '25S22');

select * from Orders

-- Tabel Pembayaran

CREATE TABLE Pembayaran (
ID_Pembayaran varchar(10) PRIMARY KEY,
Tanggal_Pembayaran date,
Metode_Pembayaran varchar(50),

```

```

Jumlah_Pembayaran int
);

select * from Pembayaran

INSERT INTO Pembayaran (ID_Pembayaran, Tanggal_Pembayaran, Metode_Pembayaran,
Jumlah_Pembayaran)
VALUES
('23P01', '2023-01-01', 'Internet Banking', 2),
('23P02', '2023-01-02', 'COD', 3),
('23P03', '2023-01-03', 'COD', 1),
('23P04', '2023-01-04', 'COD', 1),
('23P05', '2023-01-05', 'Internet Banking', 1);

select * from Pembayaran

-- Tabel Detail Pesanan

CREATE TABLE Detail_Pesanan (
    No_Pesanan INT PRIMARY KEY,
    Id_Order VARCHAR(10),
    Id_Produk VARCHAR(10),
    Nama_Pesanan VARCHAR(50),
    Kuantitas INT,
    Tanggal_Pesanan DATE,
    Harga_Satuan VARCHAR(20),
    Status_Pesanan VARCHAR(50),
    Jasa_Pengiriman VARCHAR(50),
    CONSTRAINT FK_Id_Order FOREIGN KEY (Id_Order) REFERENCES Orders(Id_Order)
)

```

```
);
```

```
select * from Detail_Pesanan
```

```
INSERT INTO Detail_Pesanan (No_Pesanan, Id_Order, Id_Produk, Nama_Pesanan, Kuantitas,  
Tanggal_Pesanan, Harga_Satuan, Status_Pesanan, Jasa_Pengiriman)
```

```
VALUES
```

```
(122, '11CP1', '11YS1', 'Baju', 2, '2023-01-01', 'Rp. 75.000', 'Sudah dibayar', 'JNT'),  
(222, '11CP2', '11YS2', 'Celana Pendek', 3, '2023-01-01', 'Rp. 70.000', 'Belum dibayar', 'JNT'),  
(322, '11CP3', '11YS3', 'Sendal Gunung', 1, '2023-01-01', 'Rp. 180.000', 'Belum dibayar', 'JNT'),  
(422, '11CP4', '11YS4', 'Sepatu', 1, '2023-01-01', 'Rp. 500.000', 'Belum dibayar', 'JNT'),  
(522, '11CP5', '11YS5', 'Hoodie', 1, '2023-01-01', 'Rp. 250.000', 'Sudah dibayar', 'JNT');
```

```
select * from Detail_Pesanan
```

```
-- Tabel Penjual
```

```
CREATE TABLE Penjual (
```

```
    ID_Penjual VARCHAR(10) PRIMARY KEY,  
    Nama_Toko VARCHAR(50),  
    Alamat_Toko VARCHAR(100)  
);
```

```
select * from Penjual
```

```
INSERT INTO Penjual (ID_Penjual, Nama_Toko, Alamat_Toko)
```

```
VALUES
```

```
('11S22', 'Carlos Shop', 'Jakarta Timur'),
```

```
('13S22', 'Mari Belanja', 'Medan'),  
('14S22', 'Jamet Store', 'Kalimantan'),  
('15S22', 'Serba ada', 'Medan'),  
('20S22', 'Your Fashion', 'Jakarta Timur');
```

```
select * from Penjual
```

```
-- Tabel Pengiriman
```

```
CREATE TABLE Pengiriman (  
    No_Resi VARCHAR(10) PRIMARY KEY,  
    Jasa_Pengiriman VARCHAR(50),  
    Tanggal_Pesanan DATE,  
    Status_Pengiriman VARCHAR(50),  
    Alamat_Pengiriman VARCHAR(100),  
    ID_Penjual VARCHAR(10),  
    FOREIGN KEY (ID_Penjual) REFERENCES Penjual(ID_Penjual)  
);
```

```
select * from Pengiriman
```

```
INSERT INTO Pengiriman (No_Resi, Jasa_Pengiriman, Tanggal_Pesanan, Status_Pengiriman,  
Alamat_Pengiriman, ID_Penjual)  
VALUES  
(JNT01', 'JNT', '2023-01-01', 'Dikemas', 'Medan', '11S22'),  
(JNT02', 'JNT', '2023-01-01', 'Dalam Perjalanan', 'Medan', '13S22'),  
(JNT03', 'JNT', '2023-01-01', 'Dalam Perjalanan', 'Tarutung', '14S22'),  
(JNT04', 'JNT', '2023-01-01', 'Dikemas', 'Balige', '15S22'),
```

```
('JNT05', 'JNT', '2023-01-01', 'Dikemas', 'Medan', '20S22');
```

```
select * from Pengiriman
```

```
-- Tabel Produk
```

```
CREATE TABLE Produk (
    ID_Produk VARCHAR(10) PRIMARY KEY,
    Nama_Produk VARCHAR(50),
    Harga VARCHAR(20),
    Kuantitas INT,
    Kategori VARCHAR(50),
    ID_Penjual VARCHAR(10),
    ID_Keranjang VARCHAR(10),
    FOREIGN KEY (ID_Penjual) REFERENCES Penjual(ID_Penjual)
);
```

```
select * from Produk
```

```
INSERT INTO Produk (ID_Produk, Nama_Produk, Harga, Kuantitas, Kategori, ID_Penjual,
ID_Keranjang)
VALUES
('11YS1', 'Baju', 'Rp. 75.000', 2, 'Pakaian', '11S22', 'P001'),
('11YS2', 'Celana Pendek', 'Rp. 70.000', 3, 'Pakaian', '13S22', 'P002'),
('11YS3', 'Sendal Gunung', 'Rp. 180.000', 1, 'Sendal', '14S22', 'P003'),
('11YS4', 'Sepatu', 'Rp. 500.000', 1, 'Sepatu', '15S22', 'P004'),
('11YS5', 'Hoodie', 'Rp. 250.000', 1, 'Pakaian', '20S22', 'P005');
```

```
select * from Produk
```

```
-- Tabel Keranjang
```

```
CREATE TABLE Keranjang (
    ID_Keranjang VARCHAR(10) PRIMARY KEY,
    ID_Pelanggan VARCHAR(10),
    Kuantitas INT,
    Catatan VARCHAR(50)
);
```

```
select * from Keranjang
```

```
INSERT INTO Keranjang (ID_Keranjang, ID_Pelanggan, Kuantitas, Catatan)
VALUES
```

```
('P001', '13S22', 2, 'Stok tersedia'),
('P002', '14S22', 3, 'Stok habis'),
('P003', '16S22', 1, 'Stok tersedia'),
('P004', '11S22', 1, 'Stok habis'),
('P005', '12S22', 1, 'Stok tersedia');
```

```
select * from Keranjang
```

```
-- Tabel Tinjauan
```

```
CREATE TABLE Tinjauan (
    ID_Tinjauan VARCHAR(10) PRIMARY KEY,
```

```
Isi_Tinjauan VARCHAR(100),
Tanggal_Tinjauan DATE,
Rating INT,
ID_Produk VARCHAR(10),
ID_Pelanggan VARCHAR(10),
FOREIGN KEY (ID_Produk) REFERENCES Produk(ID_Produk),
FOREIGN KEY (ID_Pelanggan) REFERENCES Pelanggan(ID_Pelanggan)
);
```

```
select * from Tinjauan
```

```
INSERT INTO Tinjauan (ID_Tinjauan, Isi_Tinjauan, Tanggal_Tinjauan, Rating, ID_Produk,
ID_Pelanggan)
VALUES
('001T1', 'Kualitas barang baik', '2023-01-06', 4, '11YS1', '13S22'),
('001T2', 'Kualitas barang sangat baik', '2023-01-07', 5, '11YS2', '14S22'),
('001T3', 'Kualitas barang baik', '2023-01-08', 3, '11YS3', '16S22'),
('001T4', 'Kualitas barang kurang baik', '2023-01-09', 2, '11YS4', '11S22'),
('001T5', 'Kualitas barang baik', '2023-01-10', 4, '11YS5', '12S22');
```

```
select * from Tinjauan
```

```
/*1. Used to remove a Tabel definition and all the data, indexes, triggers, constraints
and permission specifications for employees and salary_grade Tabel */
```

```
drop Table Akun
```

```
select * from Akun
```

```

/* 2. Add a data manager according to the data shown. Then, show all data manage */

INSERT INTO Detail_Pesanan (No_Pesanan, Id_Order, Id_Produk, Nama_Pesanan, Kuantitas,
Tanggal_Pesanan, Harga_Satuan, Status_Pesanan, Jasa_Pengiriman)
VALUES
(622, '11CP6', '11YS6', 'Gelang', 3, '2023-01-01', 'Rp. 15.000', 'Sudah dibayar', 'JNT'),
(722, '11CP7', '11YS7', 'Topi', 1, '2023-01-01', 'Rp. 33.000', 'Belum dibayar', 'JNT'),
(822, '11CP8', '11YS8', 'Jam', 1, '2023-01-01', 'Rp. 500.000', 'Belum dibayar', 'JNT'),
(922, '11CP9', '11YS9', 'Kaca mata', 1, '2023-01-01', 'Rp. 500.000', 'Belum dibayar', 'JNT'),
(1022, '11CP10', '11YS10', 'Boneka', 1, '2023-01-01', 'Rp. 300.000', 'Sudah dibayar', 'JNT');

select * from Detail_Pesanan

/* 3. Change the Pelanggan data which has Id_Pelanggan is 13S22! Show all
of user identity in the Pelanggan Tabel! */

UPDATE Pelanggan
SET nama = 'Andika'
WHERE Id_Pelanggan = '13S22'

SELECT * FROM Pelanggan

/* 4. Displays data on ordered items whose price is below 200,000 and has a name ending
with the letter "G" character!*/

SELECT Nama_Pesanan, Harga_Satuan
FROM Detail_Pesanan
WHERE Nama_Pesanan LIKE '%G' AND Harga_Satuan < 'Rp. 200.000';

/* 5. Change the Detail Pesanan data which has No_pesanan is 622! Show all

```

```
of Pesanan in the Detail Pesanan Tabel! */
```

```
UPDATE Detail_Pesanan  
SET No_Pesanan = '622'  
WHERE No_Pesanan = '622'  
SELECT * FROM Detail_Pesanan
```

```
/*6. Add status column to the Pelanggan Tabel which has data type varchar */
```

```
ALTER Table Pelanggan  
ADD Aktivitas VARCHAR (50);
```

```
SELECT * FROM Pelanggan
```

```
/*7. Deleting the Aktivitas column in the Pelanggan Tabel */
```

```
ALTER Table Pelanggan  
DROP COLUMN Aktivitas;
```

```
SELECT * FROM Pelanggan
```

```
/*8. Used to perform operations such as addition to Pesanan Prices */
```

```
SELECT  
    No_Pesanan,  
    Id_Order,  
    Id_Produk,  
    Nama_Pesanan,  
    Kuantitas,  
    Tanggal_Pesanan,  
    Harga_Satuan AS 'Harga Satuan',  
    CONCAT('Rp. ', FORMAT(CAST(REPLACE(REPLACE(Harga_Satuan, 'Rp. ', ','), ',', '')) +  
    10000 AS INT), 'N0') AS 'Jumlah Harga'
```

```
FROM  
Detail_Pesanan;
```

```
/*9. Multiplication between the Kuantitas column and the Harga Satuan to get the total harga*/
```

```
SELECT  
No_Pesanan,  
Id_Order,  
Id_Produk,  
Nama_Pesanan,  
Kuantitas AS 'Kuantitas',  
Tanggal_Pesanan,  
Harga_Satuan AS 'Harga_Satuan',  
CONCAT('Rp. ', FORMAT(Kuantitas * CAST(REPLACE(REPLACE(Harga_Satuan, 'Rp. ',  
'), ',', '') AS INT), 'N0')) AS 'Total Harga'  
FROM  
Detail_Pesanan;
```

```
/*10. Show No_Pesanan, Id_Produk, Nama_Pesanan, and Kuantitas that has a Kuantitas above 2  
*/
```

```
SELECT  
No_Pesanan,  
Id_Produk,  
Nama_Pesanan,  
Kuantitas  
FROM  
Detail_Pesanan  
WHERE
```

Kuantitas > 2;

/*11. Tampilkan Id_Produk dan Nama_Pesanan dari tabel Produk. Urutkan data berdasarkan id paling awal! */

SELECT

P.ID_Produk,

DP.Nama_Pesanan

FROM

Produk P

LEFT JOIN

Detail_Pesanan DP ON P.ID_Produk = DP.Id_Produk

ORDER BY

P.ID_Produk ASC;

/*1. Tampilkan Id_Pesanan dan Nama_Pesanan dari tabel Detail Pesanan*/

SELECT DISTINCT Id_Produk, Nama_Pesanan

FROM Detail_Pesanan

SELECT * FROM Detail_Pesanan

/*2. Mengembalikan Semua No_Pesanan dan Id_Produk semua akun*/

SELECT No_Pesanan, Id_Produk

FROM Detail_Pesanan

ORDER BY No_Pesanan, Id_Produk;

/* 3. Menggunakan Kombinasi Nilai di No_Pesanan dan Nama_Pesanan untuk Mengevaluasi Duplikat */

```
SELECT DISTINCT No_Pesanan, Nama_Pesanan  
FROM Detail_Pesanan
```

```
/* 4. Featuring Nama_Pesanan along with Pesanan material*/
```

```
SELECT DISTINCT Nama_Pesanan  
FROM Detail_Pesanan  
GROUP BY Nama_Pesanan
```

```
/* 5. Count Number of Unique Rows */
```

```
SELECT COUNT(DISTINCT Nama_Pesanan)  
FROM Detail_Pesanan
```

```
/*1. Menampilkan Nama dan No_telepon dari Akun yang memiliki Email 'gmail.com'*/
```

```
SELECT Nama, No_telepon  
FROM Akun  
WHERE Email IN (  
    SELECT Email  
    FROM Akun  
    WHERE Email LIKE '%@gmail.com'  
)
```

```
/*2. Menampilkan Nama dari Akun yang memiliki Id_Akun unik (tidak ada yang sama dengan  
akun lain)*/
```

```
SELECT Nama
```

```
FROM Akun  
WHERE (SELECT COUNT(*) FROM Akun AS A WHERE A.Id_Akun = Akun.Id_Akun) = 1;
```

```
/*3. Menampilkan Id_Produk dan Nama_Pesanan dari pesanan yang Status_Pesanan-nya sudah dibayar ('Sudah dibayar')*/
```

```
SELECT Id_Produk, Nama_Pesanan  
FROM Detail_Pesanan  
WHERE No_Pesanan IN (  
    SELECT No_Pesanan  
    FROM Detail_Pesanan  
    WHERE Status_Pesanan = 'Sudah dibayar'  
);
```

```
/*4. Menampilkan No_Pesanan dan Tanggal_Pesanan dari pesanan dengan Kuantitas terbanyak*/
```

```
SELECT No_Pesanan, Tanggal_Pesanan  
FROM Detail_Pesanan  
WHERE Kuantitas = (  
    SELECT MAX(Kuantitas)  
    FROM Detail_Pesanan  
);
```

```
/*5. Menampilkan No_Pesanan dan Id_Order dari pesanan yang menggunakan Jasa_Pengiriman 'JNT' dan belum dibayar*/
```

```
SELECT No_Pesanan, Id_Order  
FROM Detail_Pesanan
```

```
WHERE Status_Pesanan = 'Belum dibayar'  
AND Jasa_Pengiriman = 'JNT';
```

```
/* 1. INNER JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan  
ID_Pelanggan */
```

```
SELECT Pelanggan.Id_Pelanggan, Pelanggan.Nama AS Nama_Pelanggan, Pelanggan.Alamat  
AS Alamat_Pelanggan,  
Penjual.ID_Penjual, Penjual.Nama_Toko, Penjual.Alamat_Toko  
FROM Pelanggan  
INNER JOIN Penjual ON Pelanggan.Id_Pelanggan = Penjual.ID_Penjual;
```

```
/* 2. LEFT JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan  
ID_Pelanggan */
```

```
SELECT *  
FROM Pelanggan P  
LEFT JOIN Penjual PJ ON P.Id_Pelanggan = PJ.ID_Penjual;
```

```
/* 3. RIGHT JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan  
ID_Pelanggan */
```

```
SELECT *  
FROM Penjual  
RIGHT JOIN Pelanggan ON Penjual.ID_Penjual = Pelanggan.Id_Pelanggan;
```

```
/* 4. OUTER JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan  
ID_Pelanggan */
```

```
SELECT Pelanggan.Id_Pelanggan, Pelanggan.Nama AS Nama_Pelanggan, Pelanggan.Alamat  
AS Alamat_Pelanggan, Penjual.Nama_Toko, Penjual.Alamat_Toko  
FROM Pelanggan  
FULL OUTER JOIN Penjual ON Pelanggan.Id_Pelanggan = Penjual.ID_Penjual;
```

```
/* 5. CROSS JOIN menggabungkan tabel Pelanggan dengan tabel Penjual berdasarkan  
ID_Pelanggan */
```

```
SELECT p.Id_Pelanggan, p.Nama AS 'Nama Pelanggan', p.Alamat AS 'Alamat Pelanggan',  
pj.ID_Penjual, pj.Nama_Toko AS 'Nama Toko', pj.Alamat_Toko AS 'Alamat Toko'  
FROM Pelanggan p  
CROSS JOIN Penjual pj;
```

```
/* 1. View 1 - Daftar Detail Pesanan */
```

```
CREATE VIEW Daftar_Pesanan AS  
SELECT No_Pesanan, Nama_Pesanan, Kuantitas, Tanggal_Pesanan, Harga_Satuan  
FROM Detail_Pesanan;
```

```
SELECT * FROM Daftar_Pesanan;
```

```
/* 2. View 2 - Daftar Pelanggan */
```

```
CREATE VIEW Daftar_Pelanggan AS  
SELECT Id_Pelanggan, Nama, Alamat  
FROM Pelanggan;
```

```
SELECT * FROM Daftar_Pelanggan;
```

```
/* 3. View 3 - Detail Order Pelanggan */
```

```
CREATE VIEW Order_Pelanggan AS  
SELECT O.Id_Order, O.Nama_Produk, O.Harga, O.Kuantitas, P.Nama AS 'Nama Pelanggan'  
FROM Orders O  
JOIN Pelanggan P ON O.Id_Pelanggan = P.Id_Pelanggan;
```

```
SELECT * FROM Order_Pelanggan;
```

```
/* 4. View 4 - Tinjauan Produk */
```

```
CREATE VIEW Tinjauan_Produk AS  
SELECT T.ID_Tinjauan, T.Isi_Tinjauan, T.Tanggal_Tinjauan, T.Rating, P.Nama_Produk  
FROM Tinjauan T  
JOIN Produk P ON T.ID_Produk = P.ID_Produk;
```

```
SELECT * FROM Tinjauan_Produk;
```

```
/* 5. View 5 - Penjual dan Pengiriman */
```

```
CREATE VIEW Penjual_Pengiriman AS
SELECT PJ.ID_Penjual, PJ.Nama_Toko, PG.No_Resi, PG.Tanggal_Pesanan,
PG.Status_Pengiriman
FROM Penjual PJ
JOIN Pengiriman PG ON PJ.ID_Penjual = PG.ID_Penjual;
```

```
SELECT * FROM Penjual_Pengiriman;
```

-- Membuat fungsi untuk menghitung total harga pesanan

```
CREATE FUNCTION HitungTotalHargaFormatted(@HargaSatuan VARCHAR(20),
@Kuantitas INT)
RETURNS VARCHAR(20)
AS
BEGIN
DECLARE @Total INT
SET @Total = CAST(REPLACE(REPLACE(@HargaSatuan, 'Rp. ', ','), ',', '') AS INT) *
@Kuantitas
RETURN 'Rp.' + FORMAT(@Total, 'N0')
END
```

```
SELECT dbo.HitungTotalHargaFormatted(Harga_Satuan, Kuantitas) AS
Total_Harga_Formatted
FROM Detail_Pesanan;
```

-- Menghitung rata-rata harga dari produk

```
CREATE FUNCTION dbo.fn_HargaRataRata()
RETURNS DECIMAL(18, 2)
AS
BEGIN
    DECLARE @AvgPrice DECIMAL(18, 2);
    -- Query untuk menghitung rata-rata harga
    SELECT @AvgPrice = AVG(CAST(REPLACE(REPLACE(Harga, 'Rp. ', ','), '.', '')) AS
DECIMAL(18, 2)))
    FROM Produk;
    RETURN @AvgPrice;
END;
GO
```

```
SELECT FORMAT(dbo.fn_HargaRataRata(), 'Rp ###,###,###.00') AS 'Rata-rata Harga';
```

```
-- Menampilkan Semua Data dari Tabel Orders
```

```
CREATE PROCEDURE ShowOrders
```

```
AS
```

```
BEGIN
```

```
    SELECT * FROM Orders
```

```
END;
```

```
EXEC ShowOrders;
```

```
-- Menampilkan seluruh data dari Tabel Detail Pesanan
```

```
CREATE PROCEDURE ShowOrderDetails
```

```
AS
BEGIN
    SELECT
        No_Pesanan,
        Id_Order,
        Id_Produk,
        Nama_Pesanan,
        Kuantitas,
        Tanggal_Pesanan,
        Harga_Satuan,
        Status_Pesanan,
        Jasa_Pengiriman
    FROM
        Detail_Pesanan;
END;

EXEC ShowOrderDetails;
```